TIBCO Silver® Fabric Enabler for TIBCO BusinessEvents® Developer Guide

Software Release 3.3
June 2017

Two-Second Advantage®

TIBCO®

Important Information

Contents

TIBCO Documentation and Support Services

Documentation for this and other TIBCO products is available on the TIBCO Documentation site. This site is updated more frequently than any documentation that might be included with the product. To ensure that you are accessing the latest available help topics, visit:

https://docs.tibco.com

Product-Specific Documentation

The following documents for this product can be found on the TIBCO Documentation site:

- *TIBCO Silver® Fabric Enabler for TIBCO BusinessEvents® Installation*
- *TIBCO Silver® Fabric Enabler for TIBCO BusinessEvents® User's Guide*
- *TIBCO Silver® Fabric Enabler for TIBCO BusinessEvents® Release Notes*
- *TIBCO Silver® Fabric Enabler for TIBCO BusinessEvents® Developer's Guide*

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, contact TIBCO Support:

- For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:

  http://www.tibco.com/services/support

- If you already have a valid maintenance or support contract, visit this site:

  https://support.tibco.com

  Entry to this site requires a user name and password. If you do not have a user name, you can request one.

How to Join TIBCO Community

TIBCO Community is an online destination for TIBCO customers, partners, and resident experts. It is a place to share and access the collective experience of the TIBCO community. TIBCO Community offers forums, blogs, and access to a variety of resources. To register, go to the following web address:

https://community.tibco.com

TIBCO Silver Fabric Enabler for BusinessEvents Overview

A BusinessEvents extension looks much like a Silver Fabric Enabler. It is packaged in two Grid Libraries: one for the Enabler, and the other for the distribution.

When the SDK is installed with the TIBCO Silver® Fabric Enabler for BusinessEvents® release 3.3 or more recent, working code examples are provided in the directories: **<*TIBCO_HOME*>\sfbe\<*enabler_version*>\sdk\sample** and **<*TIBCO_HOME*>\sfbe\<*enabler_version*>\sdk\Extension-SDK**

Extensions

The extension is the Grid Library containing the Container Extension implementation. The implementation is a class that implements the `Extension` interface directly or inherits from the `BaseExtension` class.

Some plug-in modifications do not require an implementation. The following do not require an extension but rather need a distribution:

- Installer additions of plug-in-specific JAR archives to the BusinessEvents Engine classpath
- Installer addition of plug-in-specific WAR archives in a plug-in directory of the TIBCO Administrator directory

If a plug-in installer does some extra actions, such as add plug-in specific properties in `be-engine.tra`, then the plug-in needs an extension to do the same actions that the plug-in installer does. To implement and package the extension, see BusinessEvents Extension. Extension implementation and the external jar that the extension depends on are packaged in the `ds_jars` folder which is present in the directory:

TIBCO_Home\sfbe\<*enabler_version*>\sdk\Extension-SDK

A configure file named grid-library.xml is present in the *TIBCO_Home*\sfbe\<*enabler_version*>\sdk\Extension-SDK\templates

Distributions

The distribution is the Grid Library that contains the BusinessEvents plug-in used in a BusinessEvents extension Grid Library. To package the distribution, see BusinessEvents Extension Distribution Overview.

BusinessEvents Extension

A BusinessEvents extension looks much like a Silver Fabric Container. It is packaged in two Grid Libraries: one for the Container Extension, and the other for the distribution. The extension implementation and the external JAR that the extension depends on are packaged in the `ds_jars` folder. A configuration file named `grid-library.xml` is also under the root folder of the Grid Library compressed archive.

The below illustration shows the structure of the extension Grid Library compressed archive.

In this example, the `ds_jars` directory would contain the extension implementation, such as `Sample_plugin_extension_gridlib.jar`.

Extend the BaseExtension

To implement your own extension, implement an extension interface directly or inherit from `BaseExtension` class. Please refer to the example Java code below.

CommonExtension.java

```
public class CommonExtension extends BaseExtension {
@Override
public void onEvent(ExtensionEvent event) throws
ExtensionException {
if (event.getLifeCycle.equals("after-gen-machine-model")) {
// do something interesting
} else if (event.getLifeCycle.equals("after-post-install")) {
// do something interesting
} else {
// do nothing
}
}
@Override
public void onFailureEvent(ExtensionEvent event) throws
ExtensionException{
if (event.getLifeCycle.equals("after-gen-machine-model")) {
// undo something interesting
} else if (event.getLifeCycle.equals("after-post-install")) {
// undo something interesting
} else {
// undo nothing
}
}
}
```

Also you will need to prepare a `grid-library.xml` and specify the extension name, version and jar path in it. Please note that the standard version format should be four digits, and the fourth digit is the extension Hotfix version.

```
<?xml version="1.0" encoding="UTF-8"?>
<grid-library>
<grid-library-name>TIBCO_plugin_Extension_SDK_gridlib</grid-librar
y-name>
<grid-library-version>1.0.0.0</grid-library-version>
<jar-path>
<pathelement>ds_jars</pathelement>
</jar-path>
</grid-library>
```

Extension Implementation

To implement your own extension, implement the extension interface directly or inherit from the `BaseExtension class` in the Java package `com.tibco.sf.container.be.extension`. This class name is the value used in `plugin.impl in plugin.properties`. Refer to the following Java code, in the `MyFirstExtension.java` file:

```
public class MyFirstExtension implements Extension {
private final transient Logger logger =
LogUtils.forClass(this.getClass());
private BEContainer beContainer = null;
/***
* Please handle extension events in onEvent and onFailureEvent
function
* accordingly.
* Possible events are
* BEFORE_INIT,
* AFTER_INIT,
* BEFORE_START,
* AFTER_START,
* BEFORE_INSTALL,
* AFTER_INSTALL,
* BEFORE_SHUTDOWN,
* AFTER_SHUTDOWN,
* BEFORE_CLEANUP,
* AFTER_CLEANUP
***/
public void onEvent(ExtensionEvent event) throws ExtensionException {
(event.getEventName().equals(BEContainer.ContainerEvent.AFTER_
INIT.name())) {
// Write your code to do plug-in specific actions in AFTER_INIT
}
}
/***
* In case any extension throws an exception when handling an event,
the
* Container will call onFailureEvent of each plug-in that has
handled the event,
* so the extension may get a chance to rollback the work.
***/
public void onFailureEvent(ExtensionEvent event) throws
ExtensionException {
// Roll back what is done in onEvent if ant exception happens
if
(event.getEventName().equals(BEContainer.ContainerEvent.AFTER_INIT))
{
// roll back actions in onEvent
}
}
@Override
public void setContainer(Container container) throws
ExtensionException {
beContainer = (BEContainer)container;
}
@Override
public Container getContainer() throws ExtensionException {
if(beContainer == null)
throw new ExtensionException("BEContainer instance is null");
return beContainer;
}
}
```

Most plug-ins do not need to implement `onEvent()` and `onFailureEvent()` if they only add additional JAR or WAR archives in specific plug-in directories. If a plug-in needs to perform additional tasks, such as modifying the `be-engine.tra` file to add

additional properties, the plug-in needs to implement on `Event()`, most likely to do something in `AFTER_INIT`.

Building an Extension Grid Library

To build an extension Grid Library:

Procedure

1. Compile the source code of your own extension and create a JAR archive. You may need to add the libraries in the `<SDK>/lib` folder into the build path.
2. Create and configure `gridlib-library.xml`.
3. Put the JAR archive into the folder you specified in the `jar-path` element in `gridlib-library.xml`.
4. Archive the `jar-path` folder and the `gridlib-library.xml` in a compressed archive.

BusinessEvents Extension Distribution Overview

A BusinessEvents extension distribution is the archive that contains the BusinessEvents plug-in used in a BusinessEvents extension Grid Library. It is a Grid Library, consisting of all files that would normally be added to a BusinessEvents installation when a plug-in is installed, plus metadata files, packaged in a compressed archive.

Most use cases will not require a new distribution and an enabler. If you need to change the behavior of TIBCO BusinessEvents then you might require a new Distribution and an Enabler, but for 95% of cases an extension of the BaseExtension will prove more useful to accomplish developer goals.

The following illustration shows the metadata files in the top-level directory of the Grid Library.

Building an Extension Distribution

To build an extension distribution, there are five basic steps:

Procedure

1. Determine what files will be needed to install the plug-in. See Determining Required plug-in Files for more details.
2. Create a `dependency.properties` file. See Dependency.properties File for more details.
3. Create a `grid-library.xml` file. See Grid-library.xml for more details.
4. Create a `plugin.properties` file. See Plugin.properties File for more details.
5. Archive the files. See Archiving the Distribution Files for more details.

Determining Required plug-in Files

Procedure

1. To start, you must first install BusinessEvents.
2. Select the plug-in that works with the version you have installed.
3. Note all additional files and JARs that are installed when the plug-in is installed.
4. Note what files are changed in the BusinessEvents installation after the plug-in is installed so that you can later remove all files that aren't changed when the plug-in is installed. This will leave you with a set of files that could be added to any BusinessEvents installation to add the plug-in.
5. This will leave you with a set of files that could be added to any BusinessEvents installation to add the plug-in.

Dependency.properties File

If there are dependencies you must create a file named `dependency.properties`, containing the following line:

```
Sample_plugin_extension_gridlib [1.0.0, 2.0.0)
```

The name, shown above as `Sample_plugin_extension_gridlib`, is the `grid-library-name` element of the extension Grid Library. Use square brackets [] to indicate a version is inclusive, or parenthesis () to indicate that it is exclusive. The above statement means this distribution depends on `Sample_plugin_extension_gridlib` between version 1.0.0 (inclusive) and 2.0.0 (exclusive).

The `dependency.properties` file is not needed if there is no extension Grid Library for the plug-in distribution.

Grid-Library.xml File

The `grid-library.xml` file, which is located at the root of a Grid Library archive, specifies name, version, dependencies, paths, operating system, and other information about a Grid Library. The following is an example of a `grid-library.xml` for an implementation:

```
<?xml version="1.0" encoding="UTF-8"?>
<grid-library os="linux">
<grid-library-name>
Sample_plugin_extension_gridlib
</grid-library-name>
<grid-library-version>
1.0.0.0
</grid-library-version>
<jar-path>
<pathelement>ds_jars</pathelement>
</jar-path>
</grid-library>
```

Your `grid-library.xml` must contain the following elements and attributes:

- A `grid-library` element with an os attribute. The `os` attribute defines the operating system on which the distribution is running. The element and attribute with value could be defined with any of the following:

```
A grid-library element with an os attribute. The os attribute
defines the
operating system on which the distribution is running. The element
and
attribute with value could be defined with any of the following:
<grid-library os="linux">
<grid-library os="linux64">
<grid-library os="solaris">
<grid-library os="solarisX86">
<grid-library os="win32">
<grid-library os="win64">
```

If the distribution is platform-neutral the element and os attribute should be set to:

```
<grid-library os="all">
```

- A `grid-library-name` element. This defines the name of the implementation. The extension Grid Library can be named anything as long as it follows Silver Fabric Grid Library naming conventions. It is suggested to use the naming convention of *`BEPlugin Name`*`_extension_gridlib`, where *`BEPlugin Name`* is the BusinessEvents plug-in name used by TIBCO Universal installer.

- A `grid-library-version` element. This defines the version of the distribution. The standard version format should be 4 digits, with the fourth digit as the extension Hotfix version.
- A `jar-path` element. This defines the directory that will contain your implementation JAR.

For more information on the `grid-library.xml` file, see the "Using the Silver Fabric SDK" chapter of the *Silver Fabric Developer's Guide*.

Plugin.properties File

When the Container loads extensions in runtime, it will read the `plugin.properties` file of each extension. The Container may gather information such as the extension name, version, implementation class, and distribution dependencies from this file. Create a `plugin.properties` file, such as the example below.

```
product=SampleBEPlugin
version=1.0.0
plugin.impl=com.tibco.sf.container.be.extension.MyFirstPluginExten
sion
TIBCO_BusinessEvents_Distribution=[5.0.1, 5.4.1)
```

Your `plugin.properties` file must contain the following items:

- The `product` value is the BusinessEvents plug-in name.
- The `version` value is the version of the plug-in.
- The `plugin.impl` is the Java package and class name implemented in the extension. This property can be omitted if no BusinessEvents extension is needed, but the product and version properties cannot be omitted.
- The name, shown above as `TIBCO_BusinessEvents_Distribution`, is the name of the BusinessEvents distribution Grid Library the Extension is compatible with. Use square brackets [] to indicate a version is inclusive, or parenthesis () to indicate that it is exclusive. The above statement means this distribution depends on a version of BusinessEvents between 5.0.1 (inclusive) and 5.4.1 (exclusive).

Based on your latest version of distribution, the version number of the exclusive distribution will change.

Archiving the Distribution Files

Procedure

- To create the distribution Grid Library, package the above three files as well as the BusinessEvents plug-in into a compressed archive.

Examples

When the SDK is installed with the TIBCO Silver Fabric Enabler for BusinessEvents release 3.3 or more recent, working code examples are provided in the directory: *<TIBCO_HOME>*`\sfbe\3.3\sdk\sample`

There is an Extension example in the sample folder.

- `data_modeling`— A distribution for a BusinessEvents data modeling add-on.

See the `readme.txt` file in the `..\sample\data_modeling` directory for more information on building that sample distribution to run that example. You will find the ant script `build-extension.xml` in: `..\sdk\Extension-SDK\scripts`

If you would like to change the extension name and version, modify the `extension.version.properties` in the same folder before running the above build script.