TIBCO Silver® Fabric Enabler for ActiveMatrix BusinessWorks™ Developer's Guide

Software Release 3.5
June 2017

Two-Second Advantage®

Important Information

THIS 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 THIS SOFTWARE VERSION 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. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

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 "READ ME" FILES.

Contents

TIBCO Documentation and Support Services 4

Overview of BusinessWorks Extension 5

Product Overview 6

BusinessWorks Extension 7

Extension Implementation 8

Creating The Grid-library.xml File 10

Building an Extension Grid Library 11

BusinessWorks Extension Distribution 12

Building an Extension Distribution 13

Determining Required Plugin Files 14

Creating the Dependency.properties File 15

The grid-library.xml 16

The Plugin.properties File 17

Archiving the Distribution Files 18

Extension Examples 19

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 ActiveMatrix BusinessWorks™ Installation*
- *TIBCO Silver® Fabric Enabler for ActiveMatrix BusinessWorks™ User's Guide*
- *TIBCO Silver® Fabric Enabler for ActiveMatrix BusinessWorks™ Developer's Guide*
- *TIBCO Silver® Fabric Enabler for ActiveMatrix BusinessWorks™ Release Notes*
- *TIBCO Silver® Fabric Enabler for ActiveMatrix BusinessWorks™ Plug-in Distribution Quick Start 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

Overview of BusinessWorks Extension

A BusinessWorks extension distribution is the archive that contains the BusinessWorks plugin used in a BusinessWorks extension Grid Library.

An extension provides way to expose and use BusinessWorks API so they are available to your applications. Grid libraries loaded by way of the extension mechanism can by other programs.

To implement your own extension, implement the extension interface directly or inherit from the BaseExtension class in the Java package com.tibco.sf.container.bw.extension.

Product Overview

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

When the SDK is installed with the TIBCO Silver® Fabric Enabler for ActiveMatrix BusinessWorks™ release 3.5 or more recent, working code examples are provided in the directory: <TIBCO_HOME>\sfbw\3.5\sdk\sample.

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.

Note that some plugins do not require an implementation. If a plugin installer only adds plugin-specific JAR archives to the BusinessWorks Engine classpath or WAR archives in a plugin directory of the TIBCO Administrator directory, it does not need an extension; it only needs a distribution. If a plugin installer does some extra actions, such as add plugin-specific properties in bwengine.tra, then the plugin needs an extension to do the same actions as what the plugin installer does.

To implement and package the extension, see Creating a BusinessWorks Extension.

Distributions

The distribution is the Grid Library that contains the BusinessWorks plugin used in a BusinessWorks extension Grid Library. To package the distribution, see Creating an Extension Distribution.

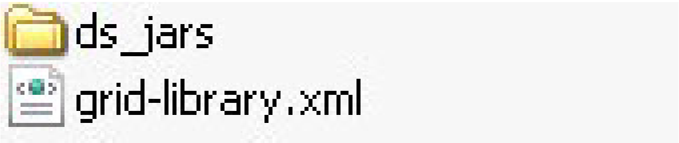
BusinessWorks Extension

A BusinessWorks 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, which 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 ZIP archive.

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



ds_jars

grid-library.xml

In the above mentioned example, the ds_jars directory contains the extension implementation, such as Sample_plugin_extension_gridlib.jar.

Extension Implementation

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

```
public class MyFirstPluginExtension implements Extension {
private final transient Logger logger = LogUtils.forClass(this.getClass());
private BWContainer bwContainer = 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(BWContainer.ContainerEvent.AFTER_ INIT.name())) {

 // Write your code to do plugin specific actions in AFTER_INIT
}
}
 /***

* In case any extension throws an exception when handling an event, the Container
will call onFailureEvent of each plugin 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(BWContainer.ContainerEvent.AFTER_INIT)) {
 // roll back actions in onEvent
}
}
 @Override
public void setContainer(Container container) throws ExtensionException {
bwContainer = (BWContainer)container;
}
@Override
 public Container getContainer() throws ExtensionException {
 if(bwContainer == null)
throw new ExtensionException("BWContainer instance is null");
 return bwContainer;
}
}
```

Most plugins do not need to implement onEvent() and onFailureEvent() if they only add additional JAR or WAR archives in specific plugin directories. If a plugin needs to perform additional tasks, such as modifying the bwengine.tra file to add additional properties, the plugin needs to implement onEvent(), and most likely to do something in AFTER_INIT.

Creating The 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>

<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:
  - <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 `BW Plugin Name_extension_gridlib`, where BW Plugin Name is the BusinessWorks plugin 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.

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

After you create a grid-library.xml file you should include it when Building an Extension Grid Library.

Building an Extension Grid Library

To build an extension Grid Library:

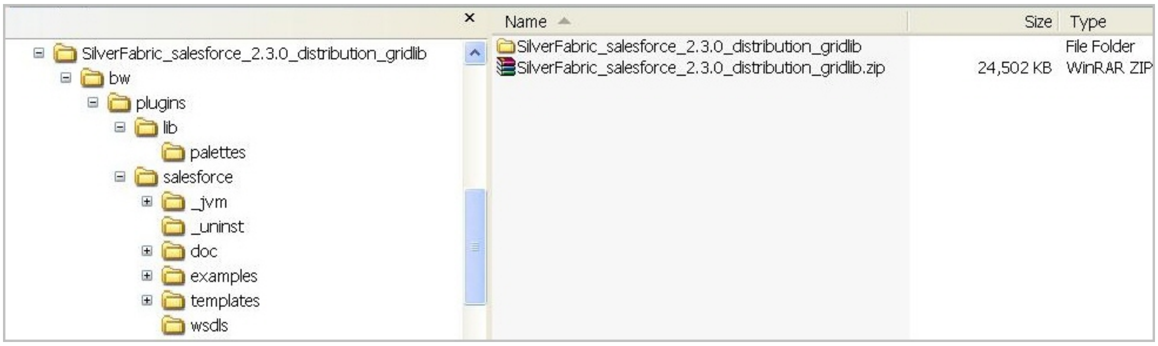
Procedure

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

BusinessWorks Extension Distribution

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

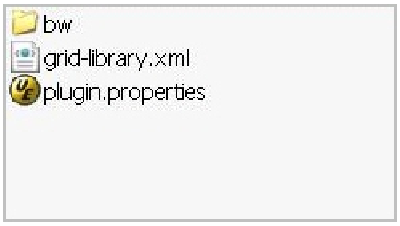
The following illustration shows the directory structure of the Salesforce example extension distribution Grid Library archive:



```
⊟ SilverFabric_salesforce_2.3.0_distribution_gridlib
    ⊟ bw
        ⊟ plugins
            ⊟ lib
                  palettes
            ⊟ salesforce
                ⊞ _jvm
                  _uninst
                ⊞ doc
                ⊞ examples
                ⊞ templates
                  wsdls
```

Name	Size	Type
SilverFabric_salesforce_2.3.0_distribution_gridlib		File Folder
SilverFabric_salesforce_2.3.0_distribution_gridlib.zip	24,502 KB	WinRAR ZIP

The following illustration shows the metadata files in the top-level directory of the Grid Library. Note that there are either two or three metadata files. If the BusinessWorks plugin needs an extension, three files are required; otherwise, only two are required.



```
bw
grid-library.xml
plugin.properties
```

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 plugin.
2. Create a `dependency.property` file.
3. Create a `grid-library.xml` file.
4. Create a `plugin.properties` file.
5. Archive the files.

Determining Required Plugin Files

Procedure

1. Install BusinessWorks.
2. Select the plugin that works with version.
3. Consider all additional files and JARs that are installed when the plugin is installed.
4. Note the delta of files changed in the BusinessWorks installation after the plugin is installed.
5. Remove all files that are not changed when the plugin is installed.

Creating the Dependency.properties File

Procedure

1. 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.
2. 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 plugin distribution.

The grid-library.xml

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 a distribution:

```
<?xml version="1.0" encoding="UTF-8"?>

<grid-library os="linux24gl23_x86">

<grid-library-name>

SampleBWPlugin_distribution

</grid-library-name>

<grid-library-version>

2.3.0.0

</grid-library-version>

</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 OS name on which the distribution is running. Possible values include `linux24gl23_x86, linux24gl23_x86, win_x86, win_x86, sol10_x86, sol10_x86, sol8_sparc, or sol8_sparc`. The os attribute can also be omitted if the distribution is platform-neutral.
- A grid-library-name element. This defines the name of the distribution. The distribution 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 `BW Plugin Name_distribution_gridlib`, where BW Plugin Name is the BusinessWorks plugin name used by TIBCO Universal installer.
- A `grid-library-version` element. This defines the version of the BusinessWorks plugin.

The 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=SampleBWPlugin

version=1.0.0

plugin.impl=com.tibco.sf.container.bw.extension.MyFirstPluginExtensio
n

TIBCO_ActiveMatrix_BW_Distribution=[5.9.2, 6.4.1)
```

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

- The `product` value - it is the BusinessWorks plugin name.
- The `version` value - it is the version of the plugin.
- The `plugin.impl` - it is the Java package and class name implemented in the extension. This property can be omitted if no BusinessWorks extension is needed, but the product and version properties cannot be omitted.
- The name - as shown above as `TIBCO_ActiveMatrix_BW_Distribution`, it is the name of the BusinessWorks distribution that is compatible with the Grid Library the Extension. 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 BusinessWorks between version 5.9.2 (inclusive) and 6.4.1 (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 BW plugin into a .zip file.

Extension Examples

When the SDK is installedwith theTIBCO Silver® Fabric Enabler for ActiveMatrix BusinessWorks™ release 3.0 or more recent, working code examples are provided in the directory: <TIBCO_HOME>\sfbw\3.5\sdk\sample.

There are three Extension examples in the `sample` folder. They include:

- `adsmartmapper` — An example of the BusinessWorks smartmapper plugin.
- `dataconversion` — An example using the BusinessWorks copybook plugin.
- `salesforce` — An example using the BusinessWorks Salesforce plugin.

See the `readme.txt` file in each example directory for more information on building extensions and distributions to run each example.