

TIBCO BusinessWorks™ Container Edition Application Monitoring and Troubleshooting

*Software Release 2.3.1
August 2017*

Document Update: September 2017

Important Information

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document contains confidential information that 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 TIBCO Software Inc.

TIBCO, TIBCO ActiveMatrix BusinessWorks, TIBCO Rendezvous, TIBCO Enterprise Message Service, TIBCO Business Studio, TIBCO Enterprise Administrator, TIBCO ActiveSpaces, TIBCO Runtime Agent, TIBCO Designer, TIBCO BusinessWorks Container Edition, TIBCO BusinessWorks Studio Container Edition and Two-Second Advantage are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Enterprise Java Beans (EJB), Java Platform Enterprise Edition (Java EE), Java 2 Platform Enterprise Edition (J2EE), and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle Corporation in the U.S. and 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.

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.

Copyright © 2001-2017 TIBCO Software Inc. All rights reserved.

TIBCO Software Inc. Confidential Information

Contents

- TIBCO Documentation4**
- Application Monitoring Overview5**
 - Application Monitoring on Cloud Foundry5
 - Configuring MySQL on Cloud Foundry5
 - Setting up TIBCO BusinessWorks™ Container Edition Application Monitoring on Cloud Foundry 7
 - Enabling Application Monitoring on Cloud Foundry 8
 - Using CUPS8
 - Using an Environment Variable9
 - Viewing Running Application 10
 - Application Monitoring on Docker 10
 - Setting Up TIBCO BusinessWorks™ Container Edition Application Monitoring on Docker 10
 - Using Docker Compose 11
 - Enabling Application Monitoring on Docker 12
 - Viewing Running Applications on Docker 13
 - Viewing Application Monitoring Dashboard 14
 - Application Statistics Collection 15
- Troubleshooting 18**
 - Connecting to the Runtime by using Secure Shell (SSH) 18
 - OSGi Commands 19

TIBCO Documentation

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

Documentation for TIBCO products is not bundled with the software. Instead, it is available on the TIBCO Documentation site.

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

- Concepts
- Installation
- Getting Started
- Application Development
- Bindings and Palettes Reference
- Samples
- Error Codes
- Migration
- Conversion
- REST Implementation
- Application Monitoring and Troubleshooting

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 including product wikis that provide in-depth information, white papers, and video tutorials. In addition, users can submit and vote on feature requests via the Ideas portal. For a free registration, go to <https://community.tibco.com>.

Application Monitoring Overview

You can run the monitoring application on the same container platform where TIBCO BusinessWorks Container Edition applications are running. TIBCO BusinessWorks Container Edition applications can be registered with monitoring application to view metrics in real-time.

Application Monitoring on Cloud Foundry

You can simply deploy an application on cloud foundry and enable the monitoring to monitor an application. The monitoring dashboard display the running application details and its statistics collection.

Procedure

1. Run the monitoring application.
2. Enable the monitoring by registering the TIBCO BusinessWorks™ Container Edition application with the monitoring application by using CUPS or environment variable. Refer [Enabling Monitoring on Cloud Foundry](#).

Configuring MySQL on Cloud Foundry

For persistence support with the monitoring application, you need to configure MySQL with either marketplace service or a user-provided service.

Creating Marketplace Service

You can check the available services by running the following command:

```
cf marketplace
```

```
xinpan-MBP15:app xinpan$ cf marketplace
Getting services from marketplace in org pcfdev-org / space pcfdev-space as admin...
OK

service      plans      description
local-volume  free-local-disk  Local service docs: https://github.com/cloudfoundry-incubator/local-volume-release/
p-mysql       512mb, 1gb      MySQL databases on demand
p-rabbitmq    standard        RabbitMQ is a robust and scalable high-performance multi-protocol messaging broker.
p-redis       shared-vm        Redis service to provide a key-value store

TIP: Use 'cf marketplace -s SERVICE' to view descriptions of individual plans of a given service.
xinpan-MBP15:app xinpan$
```

Run the following command to create a marketplace service.

```
cf create-service <SERVICE_NAME> <SERVICE_PLAN> <service_instance_name>
```

```
xinpan-MBP15:app xinpan$ cf create-service p-mysql 512mb my_test_mysql
Creating service instance my_test_mysql in org pcfdev-org / space pcfdev-space as admin...
OK
xinpan-MBP15:app xinpan$
```






MySQL is now configured with the marketplace service on the Cloud Foundry environment.

SPACE

pcfdev-space

● 1 Running
● 0 Stopped
● 0 Crashed

App (1) Services (5) Security Settings

Services			
SERVICE	NAME	BOUND APPS	PLAN
 MySQL	test_mysql	1	free - 512mb
 MySQL	my_test_mysql	0	free - 512mb
 User Provided	bwce-monitoring	0	User Provided
 User Provided	postgres	0	User Provided
 User Provided	my_mysql	0	User Provided

Creating User Provided Services

Run the following command to create the user provided service.

```
cf cups <service_instances_name> -p "host, user name, password, database"
```

```
xinpan-MBP15:~ xinpan$ cf cups t_mysql -p "host, username, password, database"

host> 127.0.0.1

username> root

password> t

database> bwadmindb
Creating user provided service t_mysql in org pcfdev-org / space pcfdev-space as admin...
OK
xinpan-MBP15:~ xinpan$
```

MySQL is now configured with the user provided service on the Cloud Foundry environment.

SPACE

pcfdev-space

2 Running
0 Stopped
0 Crashed

Apps (2) Services (5) Security Settings

SERVICE	NAME	BOUND APPS	PLAN
MySQL	test_mysql	1	free - 512mb
MySQL	my_test_mysql	0	free - 512mb
User Provided	t_mysql	0	User Provided
User Provided	bwce-monitoring	1	User Provided
User Provided	postgres	0	User Provided

Setting up TIBCO BusinessWorks™ Container Edition Application Monitoring on Cloud Foundry

The following steps describe how to set up TIBCO BusinessWorks Container Edition application on the Cloud Foundry.

Prerequisites

1. Ensure that Cloud Foundry Command Line Interface (CLI) is successfully installed and TIBCO BusinessWorks Container Edition buildpack is created in Cloud Foundry.
2. Download the TIBCO BusinessWorks Container Edition monitoring zip file, `bwce_mon.zip` from <http://edelivery.tibco.com>.
3. Ensure that MySQL service is created on Cloud Foundry. Refer [Configuring MySQL on Cloud Foundry](#).

Procedure

1. Extract the `bwce_mon.zip` file.
2. Navigate to the **bwce_mon** directory.
3. Bind the MySQL service created earlier to the monitoring application. You must configure `manifest.yml` of monitoring application to persist node registry information. You have to specify a database service and an environment variable for the MySQL database.

```

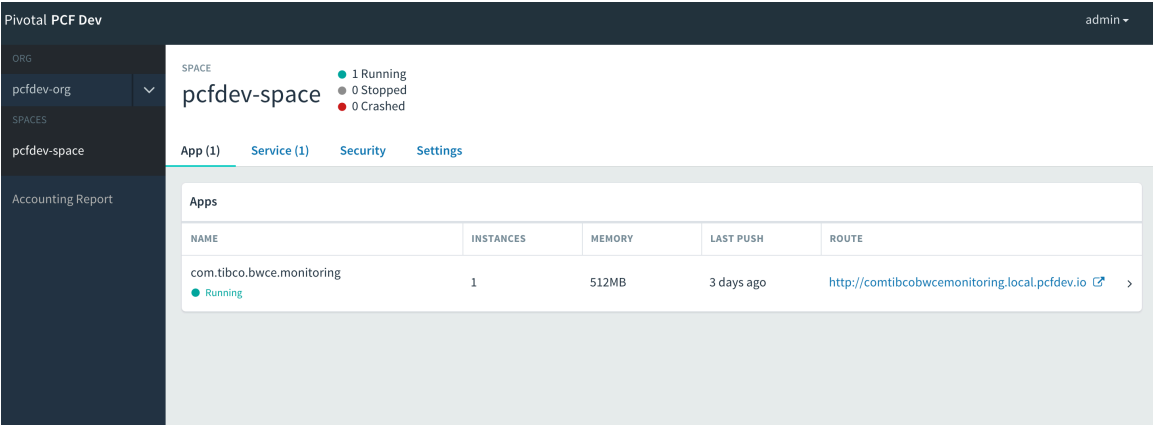
---
applications:
- name: com.tibco.bwce.monitoring
  command: node server/node-server.js
  memory: 512M
  buildpack: https://github.com/cloudfoundry/nodejs-buildpack
  services:
  - test_mysql
  env:
  persistence_DB: p-mysql

```



MySQL for Pivotal Cloud Foundry(PCF) 2.0 and later, the keyword for MySQL service is `p.mysql` and `persistence_DB` value is `p.mysql`. Similarly, MySQL for Pivotal Cloud Foundry(PCF) 1.9 and previous, the keyword for MySQL service is `p-mysql` and `persistence_DB` value is `p-mysql`.

- 4. Execute `cf push -f manifest.yml` to push the BWCE monitoring application on cloud foundry. After the BWCE application is running on Cloud Foundry, you can access the URL from a browser and monitor the application.



Enabling Application Monitoring on Cloud Foundry

Enable the monitoring for TIBCO BusinessWorks™ Container Edition application deployed on cloud foundry by using **Create User Provided Service (CUPS)** or **environment variable**.

Using CUPS

You can monitor an application by using Create User Provided Service (CUPS).

Prerequisites

Ensure that you create CUPS for TIBCO BusinessWorks Container Edition monitoring.

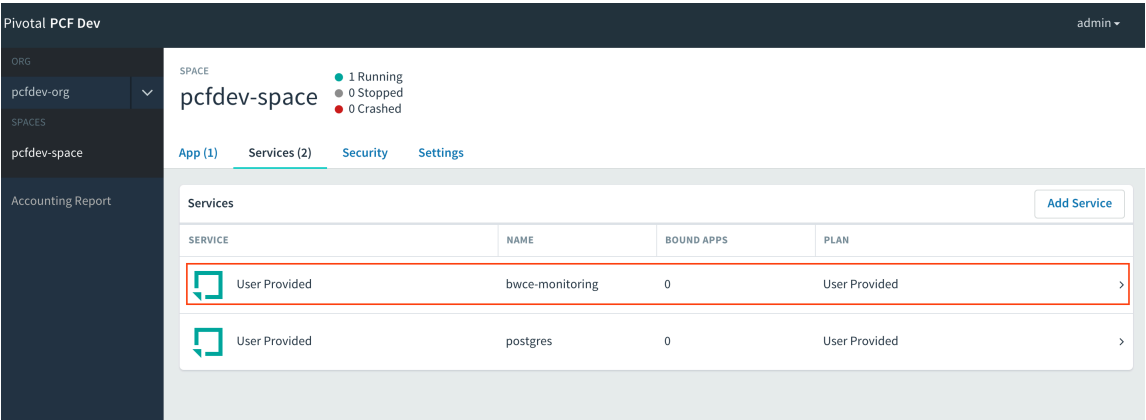
Procedure

- 1. In cf CLI, execute the command `cf cups <monitoring_app_name>-p "url"`.
For example: `http://comtibcobwcemonitoring.local.pcfdev.io`



Ensure that the name of CUPS for monitoring application should be `bwce-monitoring`.

After the command is executed, you can see the service running on PCF management web UI.



- 2. Create the `manifest.yml` file in the directory where the application EAR file is exported.
- 3. Add `bwce-monitoring` as a service in `manifest.yml`.

4. In cf CLI, run the `cf push` command to deploy the application on Cloud Foundry. After the application is deployed successfully, you can see the service running on PCF management web UI.

SPACE

2 Running

0 Stopped

0 Crashed

pcfdev-space

Apps (2)

Services (2)

Security

Settings

Apps

NAME	INSTANCES	MEMORY	LAST PUSH	ROUTE
com.tibco.bwce.monitoring	1	512MB	2 hours ago	http://comtibcobwcemonitoring.local.pcfdev.io >
tibco.bwce.sample.BookStore.application	1	1024MB	a minute ago	http://tibcobwcesamplebookstoreapplication.local... >



After the application is successfully started, the TIBCO BusinessWorks Container Edition application gets registered with the monitoring application.

Using an Environment Variable

You can monitor an application by using an environment variable.

Procedure

1. Create a `manifest.yml` in the same directory where the application EAR file is exported.
2. Set the environment variable to bind monitoring service. Add `BW_APP_MONITORING_CONFIG" <url>` as environment variable in `manifest.yml` file.

```
---
applications:
- name: RestBookStoreSample.application
  memory: 1024M
  path: tibco.bwce.sample.binding.rest.BookStore.application.ear
  timeout: 60
  buildpack: bw-buildpack
  env:
    BW_LOGLEVEL: ERROR
    BW_APP_MONITORING_CONFIG: "{\\"url\\":\\"https://monitoring.tibcopcf110.com\\"}"
```

3. In cf CLI, execute the command `cf push` to deploy the application on the Cloud Foundry.
4. After the application is deployed successfully, you can see the application running on the Cloud Foundry management web UI.

SPACE

2 Running

0 Stopped

0 Crashed

pcfdev-space

Apps (2)

Services (2)

Security

Settings

Apps

NAME	INSTANCES	MEMORY	LAST PUSH	ROUTE
com.tibco.bwce.monitoring	1	512MB	2 hours ago	http://comtibcobwcemonitoring.local.pcfdev.io >
tibco.bwce.sample.BookStore.application	1	1024MB	a minute ago	http://tibcobwcesamplebookstoreapplication.local... >

Viewing Running Application

You can monitor the running application on Cloud Foundry by accessing the routable URL.

Prerequisites

Ensure that the application is deployed on container environment.

Procedure

- Access the routable URL of monitoring application in browser to view the monitoring dashboard. You can view the following details for the running application:
 - Application name
 - Status of the application
 - Version of the application
 - Application Instances

Name	Status	Version	Instance(s)
httpgreetings.application	Running	1.0	1

Application Monitoring on Docker

You can simply deploy an application on docker and enable the monitoring to monitor an application. The monitoring dashboard displays the running application details and its statistics collection.

Setting Up TIBCO BusinessWorks™ Container Edition Application Monitoring on Docker

The following steps describe how to set up TIBCO BusinessWorks Container Edition application on Docker.

Prerequisites

Download the `bwce_mon.zip` TIBCO BusinessWorks™ Container Edition monitoring zip file, from <http://edelivery.tibco.com>.

Procedure

1. Extract the `bwce_mon.zip` file.
2. Navigate to the `bwce_mon` directory and build the docker image by running the following command.

```
docker build -t bwce/monitoring:latest .
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
bwce/monitoring	latest	018a62e2f6bb	55 seconds ago	99.8 MB
tibco/bwce	latest	f73ee3db6e78	2 days ago	352 MB
tibco/bwce	v2.3.0.23	f73ee3db6e78	2 days ago	352 MB

3. Ensure that MySQL is running and the user is created with all the privileges. You can use a standalone docker to run the monitoring application by passing the two environment variables.

- a) You must pass the two environment variables to start monitoring an application.

```
persistence_DB
DB_URL
```

- b) Run the following command.

```
docker run -p 8080:8080 -e persistence_DB="mysql" -e DB_URL="mysql://<user
name:password>@<machine:port/database> --name <containerName>
<monitoringImageName:tag>
```

```
C:\Users\bmame\Downloads\bwce_mon>docker run -p 8080:8080 -e persistence_DB="mysql" -e DB_URL="mysql://admin:admin@10.97.98.61:3306/admin" --name bwce-monitor
ing bwce/monitoring:latest
npm info it worked if it ends with ok
npm info using npm@3.10.10
npm info using node@v6.9.5
npm info lifecycle com.tibco.bwce.monitoring@2.3.0-prestart: com.tibco.bwce.monitoring@2.3.0
npm info lifecycle com.tibco.bwce.monitoring@2.3.0-start: com.tibco.bwce.monitoring@2.3.0
> com.tibco.bwce.monitoring@2.3.0 start /usr/src/app
> NODE_ENV=dev PORT=8080 node server/node-server.js

info: Initializing mysql DB...
info: Listening on port 8080
info: noderegistry table created
info: process table created
```

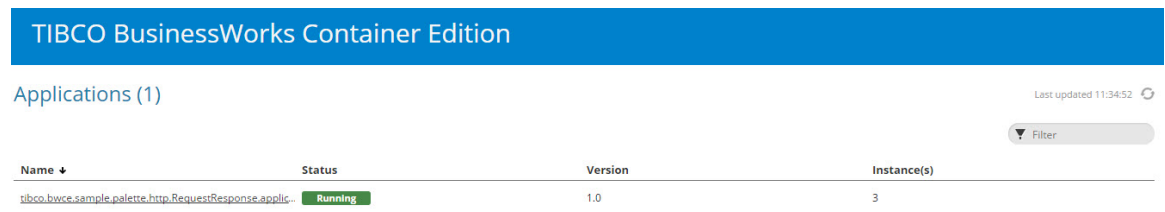
4. Run the following command to view the running container.

```
docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
d67534c36237	bwce/monitoring:latest	"npm start"	6 seconds ago	Up 5 seconds	0.0.0.0:8080->8080/tcp	bwce-monitoring

5. After the monitoring container runs successfully, you can access the monitoring UI by using following URL in the the browser:

<http://<docker-host-ip>:8080>



Using Docker Compose

You can use Docker Compose to run monitoring application along with the MySQL database on docker.

Procedure

1. Navigate to the bwce_mon directory.
2. Run the following command to build the application monitoring image.

```
docker-compose build
```

```
C:\svn\bw6mon>docker-compose up mysql_db
Pulling mysql_db (mysql:latest)...
latest: Pulling from library/mysql
9f0706ba7422: Already exists
2290e155d2d0: Already exists
547981b8269f: Already exists
2c9d42ed2f48: Already exists
55e3122f1297: Already exists
abc10bd84060: Already exists
aa37081010bb: Pull complete
aadaa7b95bc6: Downloading [====>] 5.406MB/79.61MB
8781ef2786a7: Download complete
b5c96613e09e: Download complete
3eac97813dda: Download complete
```

3. Run the following command, which downloads the MySQL image and configures the database with admin user and bwcemon database.

```
docker-compose up mysql_db
```

```
mysql_db:
  image: mysql:latest
  container_name: mon-mysql
  network_mode: bridge
  ports:
    - "3306:3306"
  environment:
    MYSQL_DATABASE: bwcemon
    MYSQL_ROOT_PASSWORD: admin
  volumes:
    - mysql_data:/var/lib/mysql
    - ./dbscripts/mysql:/docker-entrypoint-initdb.d
```

```
C:\svn\bw6mon>docker-compose up mysql_db
Pulling mysql_db (mysql:latest)...
latest: Pulling from library/mysql
9f0706ba7422: Already exists
2290e155d2d0: Already exists
547981b8269f: Already exists
2c9d42ed2f48: Already exists
55e3122f1297: Already exists
abc10bd84060: Already exists
aa37081010bb: Pull complete
aadaa7b95bc6: Downloading [====>] 5.406MB/79.61MB
8781ef2786a7: Download complete
b5c96613e09e: Download complete
3eac97813dda: Download complete
```

4. Run the following command to start the monitoring server on 8080 port.

```
docker-compose up mon_app
```

```
mon_app:
  build: .
  ports:
    - "8080:8080"
  links:
    - mysql_db
  environment:
    #DB_URL: mongodb://mongodb:27017/bwcemon
    #persistence_DB: mongo
    DB_URL: mysql://admin:admin@mon-mysql:3306/bwcemon
    persistence_DB: mysql
  network_mode: bridge
```

```
C:\svn\641bw6mon>docker-compose up mon_app
mon-mysql is up-to-date
Creating 641bw6mon_mon_app_1 ...
Creating 641bw6mon_mon_app_1 ... done
Attaching to 641bw6mon_mon_app_1
mon_app_1 | npm info it worked if it ends with ok
mon_app_1 | npm info using npm@3.10.10
mon_app_1 | npm info using node@v6.9.5
mon_app_1 | npm info lifecycle com.tibco.bwce.monitoring@2.3.0~prestart: com.tibco.bwce.monitoring@2.3.0
mon_app_1 | npm info lifecycle com.tibco.bwce.monitoring@2.3.0~start: com.tibco.bwce.monitoring@2.3.0
mon_app_1 | > com.tibco.bwce.monitoring@2.3.0 start /usr/src/app
mon_app_1 | > NODE_ENV=dev PORT=8080 node server/node-server.js
mon_app_1 |
mon_app_1 | info: Initializing mysql DB...
mon_app_1 | info: Listening on port 8080
mon_app_1 | info: table created
```

Enabling Application Monitoring on Docker

Enable the monitoring for the application using the environment variable.

Prerequisites

Ensure that the monitoring application is running on the Docker container.

Procedure

1. Create a docker file to deploy application on docker.
FROM tibco/bwce:latest MAINTAINER Tibco ADD <application name>.ear / EXPOSE 8080
2. Run the Docker terminal and navigate to the directory where the EAR and Docker file are stored.
3. Run the following command to build the application image:
docker build -t <application name> .
4. In docker run command, set an environment variable BW_APP_MONITORING_CONFIG to enable monitoring
5. Run the command in the docker terminal using docker machine IP or using link.

a) Using Application Monitoring URL

```
docker run -d -p 18065:8080 -e BW_APP_MONITORING_CONFIG='{ "url": "http://<docker-host-IP>:8080" }' <application name>
```

b) Using Link on Same Docker Host

```
docker run --link=<name or id>:alias -p 18080:8080 -e BW_APP_MONITORING_CONFIG='{ "url": "http://<alias>:8080" }' <applicationName>
```

```
xinpan-MBP15:HTTP xinpan$ docker run -P -e MESSAGE='Welcome to BWCE 2.3 !!!!!!!' --link bwce-monitoring:bwcemonitoringservice -e BW_APP_MONITORING_CONFIG='{ "url": "http://bwcemonitoringservice:8080" }' bwce-http-app
set bw.frwk.event.subscriber.metrics.enabled to true

BW_PROFILE is set to 'default.substvar'
TIBCO BusinessWorks Container Edition version 2.3.0, build V23, 2017-04-18
23:22:51.965 INFO [main] com.tibco.thor.frwk - bwappnode TIBCO BusinessWorks Container Edition version 2.3.0, build V23, 2017-04-18 initialized using logging config /tmp/tibco.home/bwce/2.3/config/logback.xml
-----
Starting AppNode framework
-----
23:23:08.000 INFO [main] com.tibco.bw.frwk.engine.BWEngine - TIBCO-BW-FRWK-300002: BW Engine [Main] started successfully.
23:23:08.947 INFO [main] com.tibco.thor.frwk - AppNode (OSGi Framework) started in 8 seconds
-----
AppNode (OSGi Framework) started in 8 seconds
-----
23:23:08.952 INFO [Framework Event Dispatcher: Equinox Container: 00396749-2026-0017-1c5e-932b80584c42] com.tibco.thor.frwk.Deployer - TIBCO-THOR-FRWK-300001: Started OSGi Framework of AppNode [standalone] in AppSpace [standalone] of Domain [standalone]
23:23:09.202 INFO [Framework Event Dispatcher: Equinox Container: 00396749-2026-0017-1c5e-932b80584c42] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300018: Deploying BW Application [docker.http.application:1.0].
23:23:09.308 INFO [Framework Event Dispatcher: Equinox Container: 00396749-2026-0017-1c5e-932b80584c42] com.tibco.thor.frwk.Application - Application bundle [docker.http.application:1.0.0.20151215223135 [423]] is resolved, but not started
23:23:09.312 INFO [Thread-21] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300008: Stopped BW Application [docker.http.application:1.0]
23:23:09.318 INFO [Framework Event Dispatcher: Equinox Container: 00396749-2026-0017-1c5e-932b80584c42] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300005: Starting BW Application [docker.http.application:1.0]
23:23:09.329 INFO [Framework Event Dispatcher: Equinox Container: 00396749-2026-0017-1c5e-932b80584c42] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300021: All Application dependencies are resolved for Application [docker.http.application:1.0]
23:23:10.434 INFO [Thread-26] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300006: Started BW Application [docker.http.application:1.0]
```

Viewing Running Applications on Docker

You can monitor the running application on Docker by accessing the Docker URL.

Prerequisites

Ensure that the application is deployed on the Docker environment.

Procedure

- Access the url `http://<docker-host-ip>:8080` to monitor the application on TIBCO BusinessWorks Container Edition monitoring web UI. You can view the following details for the running application:
 - Application name
 - Status of the application
 - Version of the application
 - Application Instances

TIBCO BusinessWorks Container Edition

Applications (1)

Last updated 12:03:34


Filter

Name	Status	Version	Instance(s)
httpgreetings.application	Running	1.0	1

Viewing Application Monitoring Dashboard

You can view App Instances, Endpoints and Processes for a running application from the application monitoring dashboard.

Procedure

- View the application status on the **Application** page. The monitoring dashboard displays the following information:
 - Total number of Application Instances, the application instances (container), and running number of instances.
 - Application version
 - REST Doc URL
 -  The REST Doc URL is shown, if the application have swagger endpoint.
 - Number of created jobs, running jobs, faulted jobs, cancelled jobs and scheduled jobs.

tibco.bwce.sample.binding.rest.BookStore.application

Running

Last updated 11:22:56

App Instances: Total (1) Running (1)

Version: 1.0

REST Doc URL: <http://demo:restbookstoreapplication@tibcopdf110.com/swagger>

Jobs: Created (1) Running (0) Faulted (0) Cancelled (0) Scheduled (0)

Stats Collection:

Process Instrumentation

ON | OFF

- On the **Applications** page, select the running application you want to view.
- To view app instances of an application, click **App Instances** tab.

App InstancesEndpointsProcesses

▼ Filter

Name	Status	Instance ▼
tibco.bwce.sample.binding.rest.BookStore.application	Running	0
tibco.bwce.sample.binding.rest.BookStore.application	Running	1

- Click the **Endpoints** tab to view endpoints exposed by the application. The type of endpoint is displayed at the top of the tab.

App Instances Endpoints Processes				
Type: REST	Filter			
EndPoint URL ↓	HTTP Methods	Client Formats	Component	Service
http://setuptibcobwcesamplebindingrest...	GET,PUT,DELETE	JSON	ComponentBooks	Book
http://setuptibcobwcesamplebindingrest...	GET	JSON	ComponentBooks	Book1
http://setuptibcobwcesamplebindingrest...	POST,GET	JSON	ComponentBooks	Books
http://setuptibcobwcesamplebindingrest...	GET,PUT,DELETE	JSON	ComponentEvents	Event
http://setuptibcobwcesamplebindingrest...	POST,GET	JSON	ComponentEvents	Events



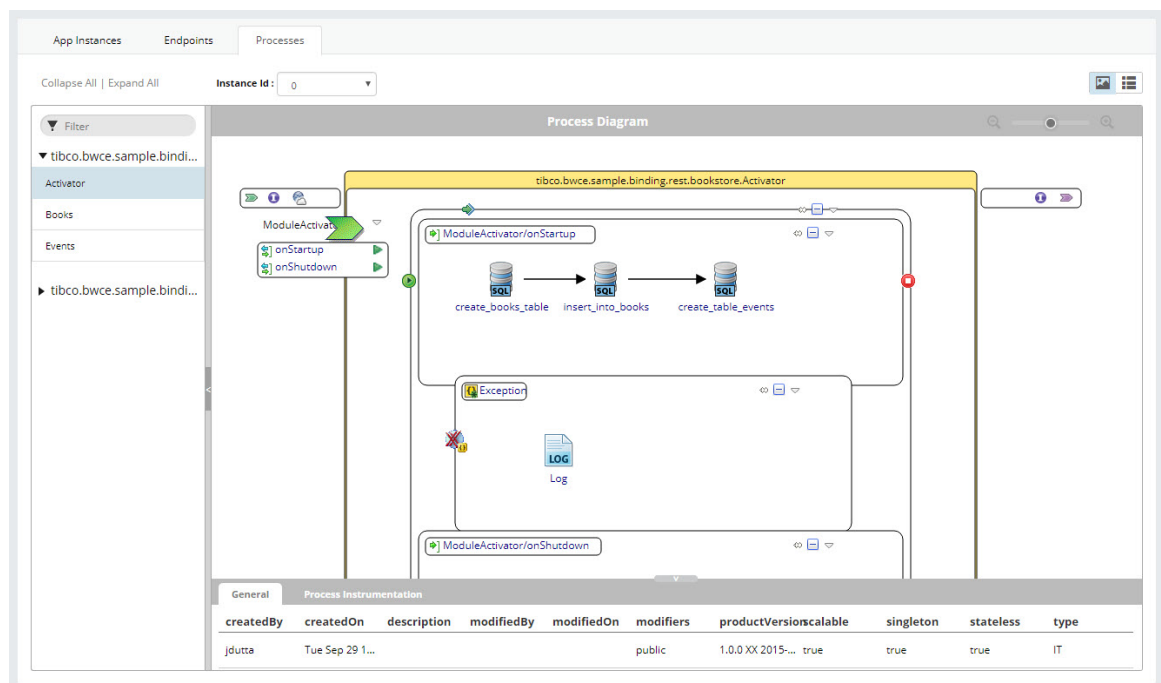
For endpoint URL in Docker based platforms, replace the container ID with the external IP on which TIBCO BusinessWorks Container Edition application is accessible.

5. Open the **Processes** tab to view application process diagram.



To view the process diagram, ensure that the version of the EAR file is 2.3.1.

- a) Use the **Instance** drop-down to select the instances of an application.
- b) You can enlarge a process diagram by clicking the **Zoom In** and **Zoom Out** button.



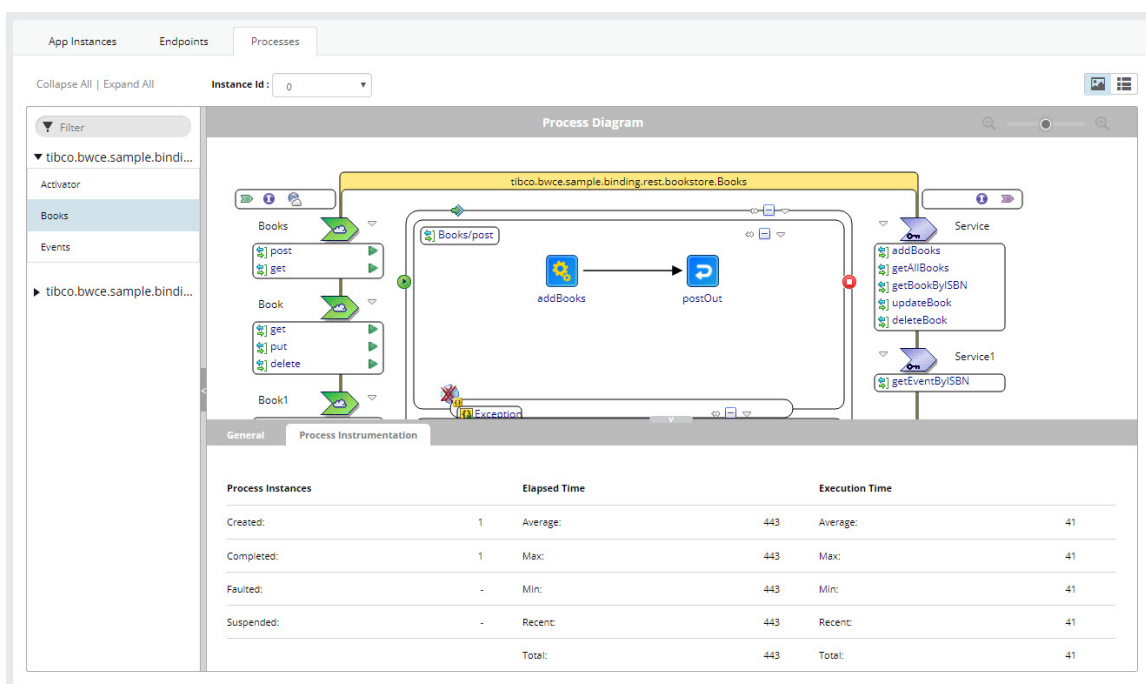
Application Statistics Collection

Application statistics collection can be enabled or disabled from the monitoring dashboard by setting the following property:


Property	Description
Process Instrumentation	To enable the monitoring of an application running on multiple app container, click the application name and click ON the Process Instrumentation property. Process Instrumentation statistics is collected for all applications.

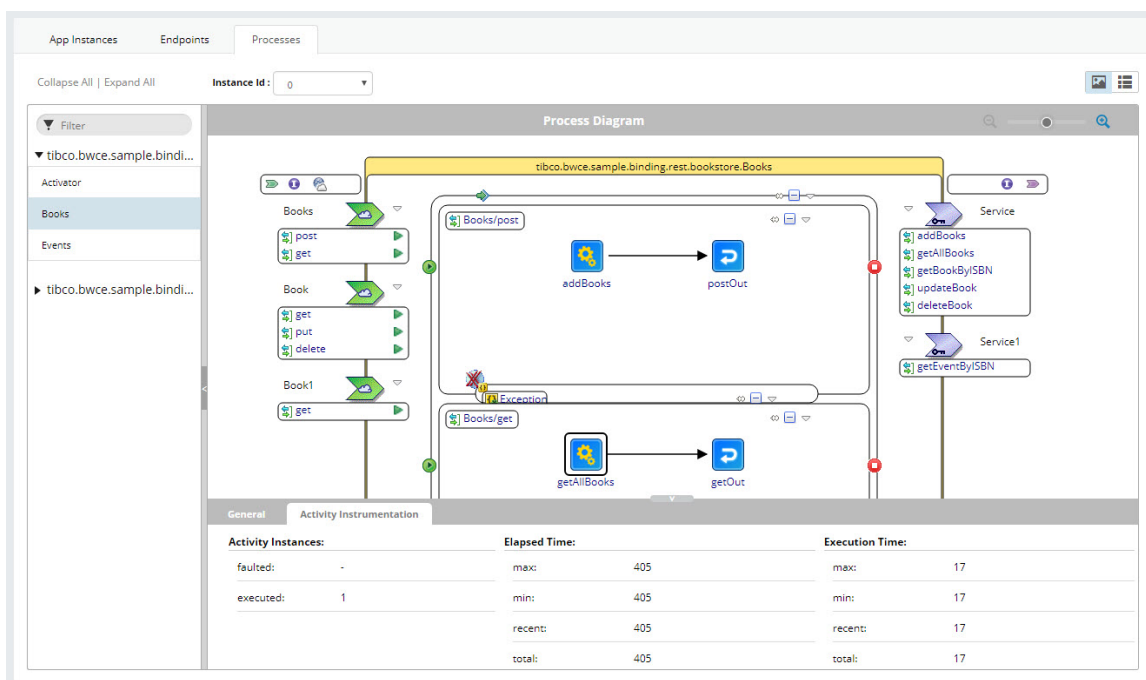
View Process Data

To view process instrumentation data, click an individual process. The process diagram, along with process instrumentation data is displayed.



View Activity Data

Select the **Processes** tab to view the process diagram. From this point you can view the activity instrumentation data by clicking an activity in the process diagram. You can view the activity instrumentation data for all activity by clicking the  icon in the upper left corner of the **Processes** tab.



App Instances

Endpoints

Processes

Collapse All | Expand All

Instance ID:

0

Filter

▼ tibco.bwce.sample.bind...

Activator

Books

Events

▶ tibco.bwce.sample.bind...

Activity Instrumentation

activityName	executed	faulted	recentStatus	ElapsedTime			ExecutionTime		
				max	min	total	max	min	total
getOut	1	-	COMPLETED	22	22	22	22	22	22
OnMessageEnd1	1	-	COMPLETED	0	0	-	0	0	-
getAllBooks	1	-	COMPLETED	368	368	368	13	13	13
pick	1	-	COMPLETED	397	397	397	2	2	2
OnMessageStart1	1	-	COMPLETED	0	0	-	0	0	-

Viewing Application Properties

You can view the application properties of an application along with its value.

TIBCO BusinessWorks Container Edition

tibco.bwce.sample.binding.rest.BookStore.application

Last updated 10:16:47

Running

App Instances: Total (1) Running (1)

Version: 1.0

Jobs: Created (2) Running (0) Faulted (0) Cancelled (0) Scheduled (0)

Stats Collection:

Process Instrumentation

ON | OFF

App Instances

Filter

0

Property

Value

tibco.bwce.sample.binding.rest.BookStore

dbDriver

org.postgresql.Driver

dbPassword

#Ah9jGFmo2aW86pEFWzAn0xpnB3jb7PmhSBL1frg

dbURL

jdbc:postgresql://54.153.40.60:5432/postgres

dbUserName

bwuser

Troubleshooting

Connecting to the Runtime by using Secure Shell (SSH)

You can connect to the TIBCO BusinessWorks™ Container Edition runtime environment by using Secure Shell (SSH).

Procedure

1. Open a terminal window to start an interactive session with the application container.
2. In the container terminal window run the following command

```
ssh -p 1122 equinox@localhost
```

3. Follow the steps in the window and also set up a new user.
The default password is `equinox`.
The new user name and password is used the next time you log in.
4. To close the secure terminal, use the command **disconnect**.

Result

A sample output for Cloud Foundry applications:

```
$ cf ssh BWCE-Performance-Test-App
vcap@ciioje9h5nd:~$ ssh -p 1122 equinox@localhost
The authenticity of host '[localhost]:1122 ([127.0.0.1]:1122)' can't be established.
DSA key fingerprint is 12:9b:20:6b:19:61:82:52:ef:7f:1b:f5:f1:ca:de:cd.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[localhost]:1122' (DSA) to the list of known hosts.
equinox@localhost's password:
Currently the default user is the only one; since it will be deleted after first
login, create a new user:
username: admin
password:
Confirm password:
roles: admin
admin@standalone>
```

OSGi Commands

After you access the OSGi console with the **telnet** command by using the Secure Shell (SSH), you can run commands to gather data about running AppNodes and applications. Refer to [Using the Secure Shell \(SSH\) to Connect to the Runtime](#).

Command Reference

- To view all commands, enter **help**
- To view command syntax, enter **help command_name** from the OSGi console, for example:

```
admin@AN1> help bw:lapi
```

The following table lists some of the commands.

OSGi Commands

Command	Description
bw:dsr	Diagnoses shared resource issues.
bw:geticon	Tests for availability of BW activity icons with a given ID and type.
bw:lais	Retrieves statistics for activities that have been executed in process(es) for the application.
bw:lapi	Retrieves information about active process instances for the application.
bw:las	Lists all instantiated activities.
bw:lat	Lists all registered activity types.
bw:lbwes	Lists all subscribers that are currently listening to BW statistics events.
bw:le	Prints information about BW engines.
bw:lec	Prints information about BW engine configurations.
bw:lendpoints	Lists endpoints exposed by the BW engine.
bw:les	Lists all instantiated EventSources.
bw:lmetrics	Prints job metrics for application(s) running on the AppNode.
bw:lpis	Prints statistics of process(es) that have been executed for the application.
bw:lr	Lists all resource details.
bw:lrhandlers	Lists all resource handlers.
bw:lrproxies	Lists all resource proxies.

Command	Description
bw:startesc	Starts collection of execution statistics for a given entity (activity/process) for application(s).
bw:stopesc	Stops execution statistics collection of given entity (process/activity) for application(s).
bw:startpsc	Starts collection of process statistics for application(s).
bw:stoppsc	Stops collection of process statistics for application(s).
bw:lapis	Prints summary of active process instance.
frwk:appnodeprocessinfo	Prints information about AppNode system processes.
frwk:dc	Delete a configuration with a given PID.
frwk:dc	Delete all configurations.
frwk:la	Print information about all applications.
frwk:lap	Print all application properties.
frwk:lb	List installed bundles matching a substring.
frwk:lb	List all installed bundles.
frwk:lcfg	Print all CAS configuration details.
frwk:lp	Print information about all known BW Processes.
frwk:ll	Print information about all libraries.
frwk:lloggers	Print all loggers currently configured on the AppNode.
frwk:lp	Print information about all known BW Processes.
frwk:pauseapp	Stop the process starters and their bindings and pause all jobs of a BW Application.
frwk:resumeapp	Start the process starters and their bindings and resume all jobs of a BW Application.
frwk:setloglevel	Sets the log level for a given logger.
frwk:startcomps	Start all process starters and their bindings of a BW Application.
frwk:startps	Start the process starters of a BW Application.
frwk:stopps	Stop the process starters of a BW Application.
frwk:stopapp	Stop an Application gracefully.

Command	Description
<code>frwk:td</code>	Print a full thread dump.



To run some of the statistics retrieval commands such as `lapi`, you must first run the `startpsc` statistics activation command.