TIBCO ActiveMatrix BusinessWorks™

Administration

Version 6.11.0 | December 2024

Copyright © 2001-2024. Cloud Software Group, Inc. All Rights Reserved.

Contents

Contents **2**

Administration Architecture Overview **8**

Getting Started **10**

Execution Modes 10

Running in Local Mode 11

Running in Enterprise Mode Using the Command Line 16

Running in Enterprise Mode Using the Admin UI 22

Core Admin Sample Scripts 28

Administrator and Agent **45**

BWAdmin 45

BWAgent 48

Configuring BWAgent 50

Creating an Agent Network 99

Accessing the BWAgent REST API with the Swagger UI 101

Using the BWAgent REST API to Return Selected Fields 104

Securing the BWAgent REST API 105

Viewing BWAgent Information 119

Restoring the File System of a BWAgent 120

Configuring the Location of the Domains Folder 121

Using BWAgent with TEA **123**

Registering BWAgent with TIBCO Enterprise Administrator 124

Autoregistering BWAgent with TIBCO Enterprise Administrator 127

Enabling and Disabling BWAgent's TIBCO Enterprise Administrator Agent Port 128

Unregistering BWAgent with TIBCO Enterprise Administrator 128

Compatibility Chart for TIBCO ActiveMatrix BusinessWorks™ and TIBCO® Enterprise Administrator 129
TEA Shell 131
Using TEA Shell Commands 132
TEA Shell Commands 133

Roles and Permissions 137

Administration Tasks and Reference 143
Managing Domains 143
Creating a Domain 143
Deleting a Domain 145
Backing Up and Restoring a Domain 147
Restoring the File System of a Domain 149
Managing AppSpaces 150
Creating an AppSpace 150
Starting an AppSpace 153
Editing an AppSpace Configuration 154
Viewing AppSpace States 156
Stopping an AppSpace 158
Deleting an AppSpace 159
Backing Up and Restoring an AppSpace 160
Restoring the File System of an AppSpace 161
Command History 162
Managing AppNodes 162
Creating an AppNode 163
Starting an AppNode 171
Editing an AppNode Configuration 172
Auto Collecting Engine Data 173
Stopping an AppNode 182
Force Shutting Down an AppNode 183
Deleting an AppNode 184

Debugging an AppNode ... 186
OSGi Commands ... 188
Backing Up and Restoring an AppNode ... 222
Restoring the File System of an AppNode ... 223
Command History ... 223
AppNode-Level Engine Properties ... 224
Enabling the OSGi Console for an AppNode ... 225
Running OSGi Commands ... 226
Running OSGi Commands from BWAdmin Command Line ... 227
Running OSGi Commands Using SSH Client ... 228
Running OSGi Commands Using HTTP Client ... 229
Disabling OSGi Commands ... 229
Managing an Application ... 230
Creating an Application ... 231
Creating an Application with Multiple Profiles ... 232
Creating an Application Archive ... 233
Uploading an Application Archive ... 234
Configuring Application Archives ... 235
Deploying an Application ... 238
Downloading an Application Archive ... 242
Editing Application and Application Instance Properties ... 244
Exporting an Application Profile ... 246
Starting an Application ... 248
Viewing Running Applications ... 249
Viewing Endpoints, Components, Processes, and Command History ... 250
Configuring a Unified Doc URL ... 252
Stopping an Application ... 255
Undeploying an Application ... 256
Starting a Component in an Application ... 257
Stopping a Component in an Application ... 260
Retrieving List of Components in an Application ... 262
Retrieving Details of a Component in an Application ... 263

Suspending and Resuming Process Instances ... 263
Backing Up and Restoring an Application ... 264
Restoring the File System of an Archive ... 265
Restoring the File System of an Application ... 266
Publishing APIs to TIBCO Cloud™ API Management ... 267

Backing Up and Restoring from the Backup ... 269

Restoring the File System of Runtime Entities ... 270

Smart Engine ... 271
Generating Reports for Engine Data ... 271
Triggers ... 278
Triggers REST API ... 282
Reports REST API ... 284
Properties REST API ... 288

Debugging ... 291
Troubleshooting BWAgent Issues ... 293
Troubleshooting Runtime Entity Issues ... 299
Troubleshooting Archive Issues ... 307
Troubleshooting Application Issues ... 309
Troubleshooting Admin UI Issues ... 312

Logging ... 317
Application Logging ... 318
Creating Separate Log Files for Each Application on the AppNode ... 321
Debugging a Specific Application on the AppNode ... 323
Supported Loggers ... 324
Backward Compatibility for Application Logging ... 325
AppNode Logging ... 325
BWAdmin Logging ... 326
Admin Message Codes ... 328

BWAgent Logging ... 343
HTTP Logging ... 345
Viewing Log Files from the Admin UI ... 347

Fault Tolerance ... 349
Application Activation Modes ... 352
Engine Persistence Modes ... 355
Configuring a Database for the Engine ... 357
Configuring the Engine for Group Persistence Mode ... 361
Configuring TIBCO FTL® as the Group Provider for Engine ... 361
Configuring EMS as the Group Provider for Engine ... 365
Configuring the Engine for FTGroup Persistence Mode ... 369
Configuring TIBCO FTL® as the FTGroup Provider for Engine ... 370
Configuring EMS as the FTGroup Provider for Engine ... 374

Engine and Job Tuning ... 379
Setting Engine and Job Tuning Properties ... 387
BWAdmin Command Line ... 388
Admin UI ... 392
Viewing Engine Properties ... 392
Engine Properties ... 393

Governance and Monitoring ... 405
Monitoring Processes ... 405
Enabling Process Monitoring ... 405
Configuring using REST ... 410
Configuring using UDP ... 412
Configuring using FTL ... 414
Configuring with CSV ... 417
Application Statistics Collection ... 418
Application Metrics ... 419
Process Statistics ... 420
Enabling and Disabling Process Statistics ... 420

Viewing Collected Statistics ... 422
Process Execution Statistics ... 426
Integrating Execution Statistics Collection Using Logback ... 428
Enabling and Disabling Auditing Events ... 432
Applying Security Policies ... 434
Enabling the Governance Agent Using the Admin UI ... 434
Enabling the Governance Agent Using an AppSpace Configuration File ... 436

OpenTelemetry ... 440
Traces ... 443
Custom Tags for OpenTelemetry ... 447
OpenTelemetry Tags from Palettes ... 447
Metrics ... 461

List of Ports ... 463

TIBCO Documentation and Support Services ... 465

Legal and Third-Party Notices ... 467

Administration Architecture Overview

Applications are deployed into runtime environments and managed using the BWAdmin utility.TIBCO ActiveMatrix BusinessWorks™ provides a flexible framework that you can use to scale your runtime environment as needed. The runtime also provides an option to run the engine so that the risk of a single point of failure when running an application is reduced.

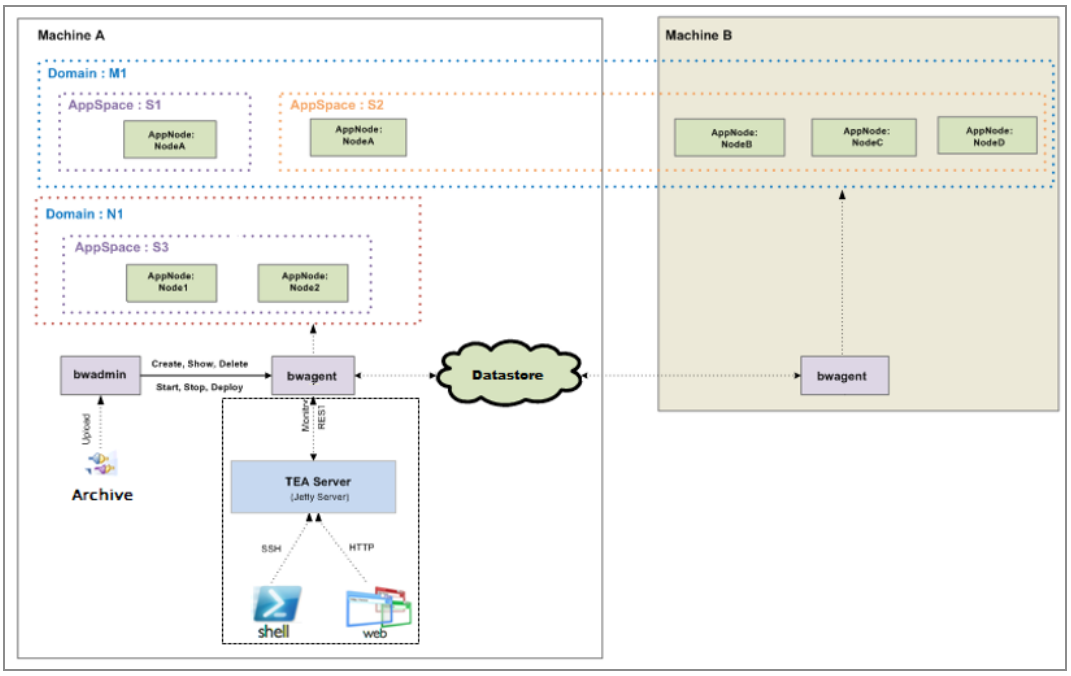
These are the key administrative components:

- An Application Archive is the deployment unit for an application that is generated in TIBCO Business Studio™ for BusinessWorks™.
- A domain is a logical group that provides an isolated environment for applications and their resources to reside.
- An AppSpace is a group of one or more AppNodes, which are runtime entities that host ActiveMatrix BusinessWorks™ applications. AppSpaces are contained within a domain. One or more applications can be deployed to an AppSpace.
- An AppNode is a runtime entity that hosts applications. AppNodes are contained in an AppSpace.
- The BWAgent is a daemon that runs on every ActiveMatrix BusinessWorks installation. When multiple installations across machines are configured as a network, the BWAgents interact with each other using a datastore. They also synchronize the data from the datastore with the local file system.

The Administration Architecture illustration below shows an example of runtime entities created across two BWAgents in a network. In the illustration, domain M1 spans two machines, Machine A and Machine B. Domain N1 is on Machine A only. Domain M1 contains two AppSpaces, S1 and S2. AppSpace S2 spans both machines. The BWAgent on Machine A is configured to interact with the BWAgent on Machine B through the datastore.

The Admin UI is a web UI that runs in TIBCO® Enterprise Administrator (TEA). Using the Admin UI is optional. To enable the Admin UI, the BWAgent must be registered with a running TEA server. In the following Administration Architecture illustration, the BWAgent on Machine A is registered with the TIBCO Enterprise Administrator (TEA) server. If the registered BWAgent becomes unavailable, the connection between the TEA server and the agent network is automatically recovered. The BWAgent on Machine B automatically registers with the server.

Administration Architecture



The runtime entities manifest as a hierarchical folder structure on the local file system. Every action performed on the runtime entities results in an update to the file system. The location of the default domains folder in the local file system can be changed by editing the *BW_HOME*`/domains/DomainHomes.properties` file.

When runtime entities span machines, the BWAgent synchronizes the data from the datastore with the local file system. At any given point in time, the data in the file system is the source of truth. This ensures that in case of a failure in the communication channel, the runtime is not affected as it refers to the data on the local file system.

Note: In your production environment, ensure you are using an external database and either TIBCO FTL® or TIBCO Enterprise Message Service™ (EMS) for data persistence and communication transport.

For more information about administration concepts, see the TIBCO ActiveMatrix BusinessWorks™ Concepts guide.

Getting Started

Deploy and manage applications created in TIBCO Business Studio™ for BusinessWorks™ using the BWAdmin console or the Admin UI, and the BWAgent.

The BWAdmin console and the BWAgent are executables in the `bin` folder of the product installation. For more information about the BWAdmin console and the BWAgent, see Administrator and Agent.

For information about the Admin UI, see Using the Admin UI.

There are several ways to deploy an application:

- BWAdmin: For more information about deploying with BWAdmin, see Deploying an Archive.
- Admin UI: For more information about deploying with the Admin UI, see Deploying an Archive.
- Deployment servers in TIBCO Business Studio for BusinessWorks: For more information, see Deploying an Application in *TIBCO ActiveMatrix BusinessWorks™ Application Development*.
- Rest API

Important: In this document, *BW_HOME* points to *`TIBCO_HOME`*`\bw\n.n`.

Execution Modes

The execution mode is set using the BWAdmin command-line console or in the BWAgent's configuration file.

The execution mode determines whether the BWAdmin communicates with the BWAgent. There are two modes: local and enterprise. The default mode is set to local, meaning that there is no communication with the BWAgent.

Local Mode

In Local Mode, BWAdmin modifies the local file system directly instead of delegating the work to a BWAgent. Local mode does not provide data storage and runtime entities are created in the file system. This mode is useful for developers during development and testing cycles. For more information, see Running in Local Mode.

Enterprise Mode

In Enterprise Mode, BWAdmin communicates with BWAgent. BWAgents can communicate across machines and can be configured to form a BWAgent network. Instead of working on the file system directly, BWAdmin sends commands to the BWAgent. The BWAgent dispatches the command to a targeted agent. That agent then completes the command on the local file system. For more information, see Running in Enterprise Mode Using the Command Line. In Enterprise Mode, the BWAgent can be registered with the TEA server.

To change the mode, navigate to `BW_HOME\bin` (Windows) or `${BW_HOME}/bin` (Unix) and issue the following command: `bwadmin mode local` (to switch to Local Mode) or `bwadmin mode enterprise` (to switch to Enterprise Mode).

Changing the mode sets the `bw.admin.mode` property in the BWAgent configuration file. This file is called `bwagent.ini` and is available in the `BW_HOME\config` folder (Windows) or `${BW_HOME}/config` folder (Unix).

Running in Local Mode

Local mode allows application testing and debugging on the local file system.

This procedure shows you how to create runtime entities and deploy and run an application using BWAdmin local mode.

You can perform the following actions:

- Set the BWAdmin mode
- Create a domain, AppSpace, and AppNode
- Upload an application archive
- Start the AppSpace
- Deploy and start the uploaded application

Note: The runtime entities created in local mode are not visible to BWAgents when they are started.

Procedure

1. In a terminal, navigate to the following paths:
   - Windows:

     ```
     BW_HOME\bin
     ```

   - Unix:

     ```
     ${BW_HOME}/bin
     ```

2. Set the BWAdmin mode to local.
   - Windows:

     ```
     BW_HOME\bin>bwadmin mode local
     ```

   - Unix:

     ```
     [root@BW_HOME bin]# ./bwadmin mode local
     ```

3. Create a domain. For more information, see Creating a Domain.
   - Windows:

     ```
     BW_HOME\bin>bwadmin create domain D1
     ```

   - Unix:

     ```
     [root@BW_HOME bin]# ./bwadmin create domain D1
     ```

4. Show the domain.
   - Windows:

```
BW_HOME\bin>bwadmin show domain
```

- Unix:

```
[root@BW_HOME bin]# ./bwadmin show domain
```

5. Create an AppSpace in the domain. For more information, see Creating an AppSpace.
   - Windows:

```
BW_HOME\bin>bwadmin create -d D1 appspace AS1
```

   - Unix:

```
[root@BW_HOME bin]# ./bwadmin create -d D1 appspace AS1
```

6. Create an AppNode in the AppSpace. When creating an AppNode, specify the HTTP management port that allows communication with the AppNode.
   - Windows:

```
BW_HOME\bin>bwadmin create -domain D1 -appspace AS1 -httpPort
8060 appnode AN1
```

   - Unix:

```
[root@BW_HOME bin]# ./bwadmin  create -domain D1 -appspace AS1
-httpPort 8060 appnode AN1
```

   The HTTP management port must be unique across all defined AppNodes on a machine. If the specified port is already in use, an error occurs, and the AppNode is not created.

   For more information, see Creating an AppNode.

7. Use the `show` command for the AppSpace after you have created the AppNode.

```
BW_HOME\bin>bwadmin show -domain D1 -appspace AS1 appnodes
```

8. Upload an application archive into the domain. The following command uploads the

Bookstore sample application archive. Use forward slash in the Windows command line.

- Windows:

```
BW_HOME\bin>bwadmin upload -domain D1
../samples/core/admin/ears/bookstore/tibco.bw.sample.binding.re
st.BookStore.application_1.0.0.ear
```

- Unix:

```
[root@BW_HOME bin]# ./bwadmin upload -domain D1
../samples/core/admin/ears/bookstore/tibco.bw.sample.binding.re
st.BookStore.application_1.0.0.ear
```

For more information, see Uploading an Archive.

9. Show that the application archive was uploaded.
   - Windows:

```
BW_HOME\bin>bwadmin show -domain D1 archives
```

   - Unix:

```
[root@BW_HOME bin]# ./bwadmin show -domain D1 archives
```

10. Start the AppSpace. This starts the AppNode in the AppSpace.
    - Windows:

```
BW_HOME\bin>bwadmin start -domain D1 appspace AS1
```

    - Unix:

```
[root@BW_HOME bin]# ./bwadmin start -domain D1 appspace AS1
```

    For more information, see Starting an AppSpace.

11. Deploy the application into the AppSpace. This deploys the application to the AppNode.

- Windows:

```
BW_HOME\bin>bwadmin deploy -domain D1 -appspace AS1
tibco.bw.sample.binding.rest.BookStore.application_1.0.0.ear
```

- Unix:

```
[root@BW_HOME bin]# ./bwadmin deploy -domain D1 -appspace AS1
tibco.bw.sample.binding.rest.BookStore.application_1.0.0.ear
```

For more information, see Deploying an Application.

12. See the deployed application using the `show` command.
    - Windows:

```
BW_HOME\bin>bwadmin show -domain D1 application
```

    - Unix:

```
[root@BW_HOME bin]# ./bwadmin show -domain D1 application
```

13. Start the application. Each uploaded application maintains a version. A version number is required to start and stop the application.
    - Windows:

```
BW_HOME\bin>bwadmin start -d D1 -appspace AS1 application
tibco.bw.sample.binding.rest.BookStore.application 1.0
```

    - Unix:

```
[root@BW_HOME bin]# ./bwadmin start -d D1 -appspace AS1
application
tibco.bw.sample.binding.rest.BookStore.application 1.0
```

    To find the version number, you can use the `show` command, for example: `bwadmin show -domain D1 application`

    For more information, see Starting an Application.

    Use the `-csv` command to print the table content as a comma-separated value table.

The first row contains the headers and applies to domains, AppSpaces, AppNodes, Applications, Archive, Archives, Machines, and Installations. For example, use `bwadmin [admin@d1]> show -csv apps` to show all applications in the CSV format.

14. Optionally, stop and undeploy the application, stop the AppSpace, and delete the entities (archive, AppNode, AppSpace, and domain).

Result

You used BWAdmin in local mode to create a domain (D1), an AppSpace (AS1), and an AppNode (AN1). You uploaded an application archive to the domain, deployed the application, and started and stopped the application. Spend some time experimenting with BWAdmin commands. For more information about domains, AppSpaces, AppNodes, and applications, see Administration Tasks and Reference.

Running in Enterprise Mode Using the Command Line

In Enterprise Mode, BWAgents can communicate across machines and can be configured to form a BWAgent network.

This procedure shows you how to set up a network using BWAgents on two machines and use BWAdmin to create runtime entities across machines.

Enterprise mode requires a data persistence and communication transport layer to keep BWAgents in sync across machines. By default, TIBCO FTL® is used for communication transport and the external database for data persistence. The software also provides the option of using TIBCO Enterprise Message Service™ (EMS) for communication transport.

For more information about configuring BWAgent, see Configuring BWAgent.

The following example uses TIBCO FTL as the default configuration.

Before you begin

Install the software on machines M1 and M2 (noted in the instructions). Make a note of the host name or IP address for each machine.

Procedure

1. Configure the BWAgent on machine M1.

   For more information about configuring BWAgent with the TIBCO FTL database, see Database with TIBCO FTL®

2. Repeat the configuration for the BWAgent on machine M2.

   Ensure that the same database, TIBCO FTL, and network name are used for M1 and M2.

3. Start the BWAgent on M1.

   a. Open a terminal on M1 and navigate to the *BW_HOME*`\bin` folder (Windows) or `[root@`*BW_HOME*` bin]#` (Unix).

   b. Type `bwagent` (Windows) or `./bwagent` (Unix).

      The BWAgent starts.

4. Start the BWAgent on M2.

   a. Open a terminal on M2 and navigate to *BW_HOME*`\bin` folder (Windows) or `${`*BW_HOME*`}/bin` (Unix).

   b. Type `bwagent` (Windows) or `./bwagent` (Unix).

      The BWAgent starts.

5. Start the BWAdmin console on the M1 machine and show the BWAgents.

   - Windows:

     ```
     BW_HOME\bin>bwadmin show agents
     ```

   - Unix:

     ```
     [root@BW_HOME bin]# ./bwadmin show agents
     ```

   The `TEA Server URL`, `Registered TEA Agent` and `Auto Registration` settings are important when you are using the Admin UI. These settings can be ignored for this example. For more information about the Admin UI, see Running Applications in Enterprise Mode using the Admin UI and Using the Admin UI.

6. Now the BWAgents on machines M1 and M2 are communicating with each other. You can use the BWAdmin console on M1 to create a domain on M2 by specifying the BWAgent in the command line.

- Windows:

```
BW_HOME\bin>bwadmin create -agent M2 domain D1M2
```

- Unix:

```
[root@BW_HOME bin]# ./bwadmin create -agent M2 domain D1M2
```

7. On M1, use the `show domains` command to show the domain.
   - Windows:

```
BW_HOME\bin>bwadmin show domains
```

   - Unix:

```
[root@BW_HOME bin]# ./bwadmin show domains
```

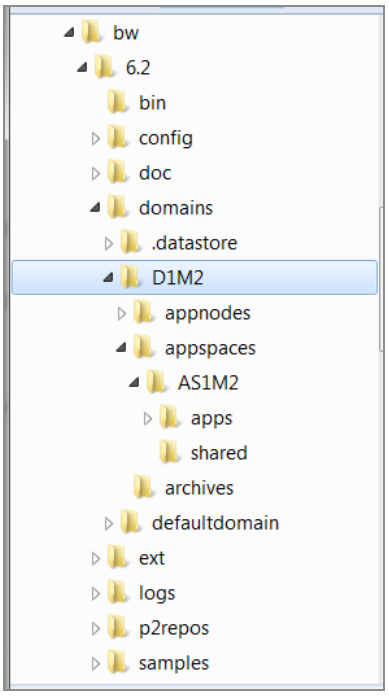
8. Create an AppSpace on M2 using BWAdmin on M1.
   a. Create an AppSpace on M2 by specifying the BWAgent. Two AppNodes are specified. The `-minNodes` option specifies the number of nodes in the AppSpace. The AppSpace cannot be started until this value is met. For more information about AppSpaces and AppNodes, see the Administration Concepts topic in the *TIBCO ActiveMatrix BusinessWorks™ Concepts* guide.
      - Windows:

```
BW_HOME\bin>bwadmin create -agent M2 -domain D1M2 -
minNodes 2 appspace AS1M2
```

      - Unix:

```
[root@BW_HOME bin]# ./bwadmin create -agent M2 -domain
D1M2 -minNodes 2 appspace AS1M2
```

   b. View the file system on machine M2 to verify that the AppSpace was created. The example below shows the file system on a Windows machine.



Each runtime entity is created in the file system. It is critical that all runtime entities are managed using BWAdmin so that they are in sync with the datastore.

9. On machine M1, create the two AppNodes for the AppSpace AS1M2, specifying BWAgent M2 for the AppNodes. The HTTP management port must be unique. A list of defined AppNodes for a given domain, including port numbers, is available with the `show` command: `show -d <DomainName> appnodes`

> **Note:** When an AppNode is created, an optional port for the OSGi console can be specified to monitor the AppNode (Only enable this port for troubleshooting purposes.). For more information, see Enabling the OSGi Console for an AppNode.

Windows:

```
BW_HOME\bin>bwadmin create -agent M2 -domain D1M2
-appspace AS1M2 -httpPort 8070 appnode AN1M2
```

```
BW_HOME\bin>bwadmin create -agent M2 -domain D1M2
```

```
-appspace AS1M2  -httpPort 8071 appnode AN2M2
```

Unix:

```
[root@BW_HOME bin]# ./bwadmin create -agent M2 -domain D1M2
-appspace AS1M2 -httpPort 8070 appnode AN1M2
```

```
[root@BW_HOME bin]# ./bwadmin create -agent M2 -domain D1M2
-appspace AS1M2  -httpPort 8071 appnode AN2M2
```

The `-httpPort` option is case-sensitive.

10. From the BWAdmin console on M1, upload an application archive into the domain on M2.

    Windows:

    ```
    BW_HOME\bin>bwadmin upload -domain D1M2
    ../samples/core/admin/ears/bookstore/tibco.bw.sample.binding.rest.B
    ookStore.application_1.0.0.ear
    ```

    Unix:

    ```
    [root@BW_HOME bin]# ./bwadmin upload -domain D1M2
    ../samples/core/admin/ears/bookstore/tibco.bw.sample.binding.rest.B
    ookStore.application_1.0.0.ear
    ```

11. From M1, start the AppSpace on M2. This starts the AppNodes in the AppSpace on M2.

    Windows:

    ```
    BW_HOME\bin>bwadmin start -domain D1M2 appspace AS1M2
    ```

    Unix:

    ```
    [root@BW_HOME bin]# ./bwadmin start -domain D1M2 appspace AS1M2
    ```

12. From the BWAdmin console on M1, verify that the AppNodes are running:

Windows:

```
BW_HOME\bin>bwadmin show -domain D1M2 -appspace AS1M2 appnodes
```

Unix:

```
[root@BW_HOME bin]# ./bwadmin show -domain D1M2 -appspace AS1M2
appnodes
```

13. Use BWAdmin on M1 to stop the AppSpace on M2. This stops AppNodes AN1M2 and AN2M2.

    Windows:

    ```
    BW_HOME\bin>bwadmin stop -domain D1M2 appspace AS1M2
    ```

    Unix:

    ```
    [root@BW_HOME bin]# ./bwadmin stop -domain D1M2 appspace AS1M2
    ```

14. Back up the domain. The `backup` command exports the persisted state of runtime entities into a command file. This command file can be used to recreate the environment. For more information, see Backing Up and Restoring from the Backup.

    The `backup` command requires the name of the specific entity to be backed up (domain, agent, AppSpace, or AppNode) as well as the path to a destination file. In this example (Windows), D1M2 is backed up.

    ```
    BW_HOME\bin>bwadmin backup -s backup.cmd domain D1M2
    ```

    Note: The BWAdmin `backup` command and the BWAdmin `restore` command are not complimentary. The `backup` command exports the current state of the environment to a command file. The `restore` command restores the file system of a BWAgent to the state of the persistent datastore. For more information, see Backing Up and Restoring from the Backup and Restoring the File System of Runtime Entities.

Result

You set up a network with two BWAgents on two machines. You used the BWAdmin console on one machine to create runtime entities on the other machine. You uploaded an

application archive to the domain and started the AppSpace. You also backed up the environment.

You can continue experimenting by adding additional machines to the network, adding more runtime entities, or deploying the archive. For this, start the AppSpace again.

Then, force delete the domain using BWAdmin on either machine with the following BWAdmin command: `delete -force domain D1M2`

After you delete the domain, you can recreate the environment from the backup by feeding the backup command file to BWAdmin, for example: `bwadmin -f backup.cmd` (Windows).

To exit the BWAgent, type ^C (this may take a few seconds). At the command line, type `bwagent stop` to stop the agent completely.

Running in Enterprise Mode Using the Admin UI

Use the Admin UI to manage and monitor runtime entities.

The Admin UI is a web UI that runs in TIBCO® Enterprise Administrator (TEA). To enable the Admin UI, the BWAgent must be registered with a running TEA server.

This procedure shows you how to create runtime entities and deploy and run an application using the Admin UI.

You can perform the following actions:

- Register the TEA agent with the BWAgent
- Open the Admin UI
- Create a domain, AppSpace, and AppNode
- Upload an archive
- Start the AppSpace
- Deploy and start the uploaded application

Procedure

1. Install TEA and start the TEA server.
    - Windows:

```
TEA_HOME\2.0\bin>tea.exe
```

- Unix:

```
[root@TEA_HOME bin]# ./tea.sh
```

2. In a terminal, navigate to the following location:
   - Windows:

```
BW_HOME\bin
```

   - Unix:

```
BW_HOME/bin
```

3. Set the BWAdmin mode to enterprise.
   - Windows:

```
BW_HOME\bin>bwadmin mode enterprise
```

   - Unix:

```
[root@BW_HOME bin]# ./bwadmin mode enterprise
```

4. Open a new terminal and navigate to *BW_HOME*\bin for Windows or ${*BW_HOME*}/bin for Unix. Register the BWAgent TEA agent with the TEA server. This allows the BWAgent to be available to the TIBCO Enterprise Administrator server. The URL to the TEA server is required in the command. The URL is available from the terminal where the TEA server was started.
   - Windows:

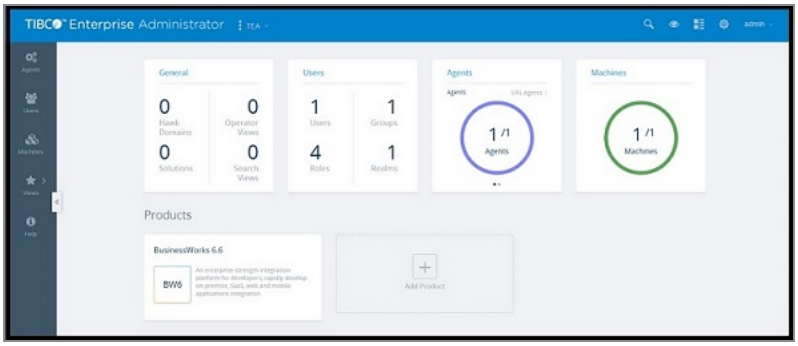
```
BW_HOME\bin>bwadmin registerteaagent http://M1:8777/
```

   - Unix:

```
[root@BW_HOME bin]# ./bwadmin registerteaagent http://M1:8777/
```

5. Open a web browser and go to the TEA URL. Sign in using `admin` for the username and `admin` as password.

   BusinessWorks is displayed in the **Products** list.



6. Click the BusinessWorks icon to go to ActiveMatrix BusinessWorks. The **Domain Management** page is displayed. If you completed the steps in the "Running in Enterprise Mode Using the Command Line", you see a domain listed on the **Domain Management** page. Otherwise the page is empty.

   a. Click **Create Domain** to open the **Create Domain** dialog.

   b. Enter the domain name in the **Name** field.

   c. Choose the BWAgent registered with the TEA server from the **Agent** dropdown.

   d. Click **Create** to create the domain.

   The domain is created and displayed on the Domain Management page.

7. Click the domain name to open the domain.

8. Add an AppSpace to the domain.

   a. Click the **AppSpaces** icon to open the AppSpaces page.

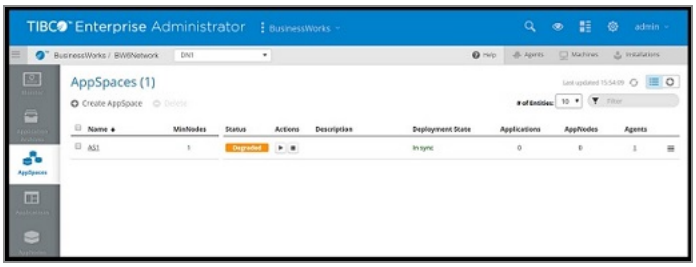
   b. Click **Create AppSpace**.

   c. In the Create AppSpace dialog, enter the name of the AppSpace in the **Name** field. Accept the value of 1 in the **MinNodes** field.

   d. Select the agent registered with the TEA server and click **Create**.

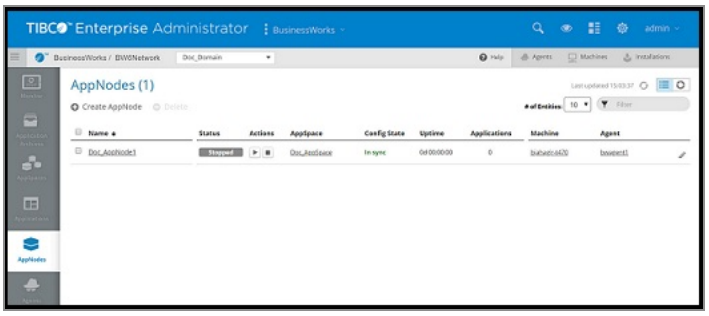
   The AppSpace is created. A success notification is displayed at the top of the page. Notice that the AppSpace status is displayed as **Degraded**, because there are no AppNodes yet for the AppSpace. This changes to **Stopped** when an AppNode is

added to the AppSpace. The AppSpaces page looks similar to this:



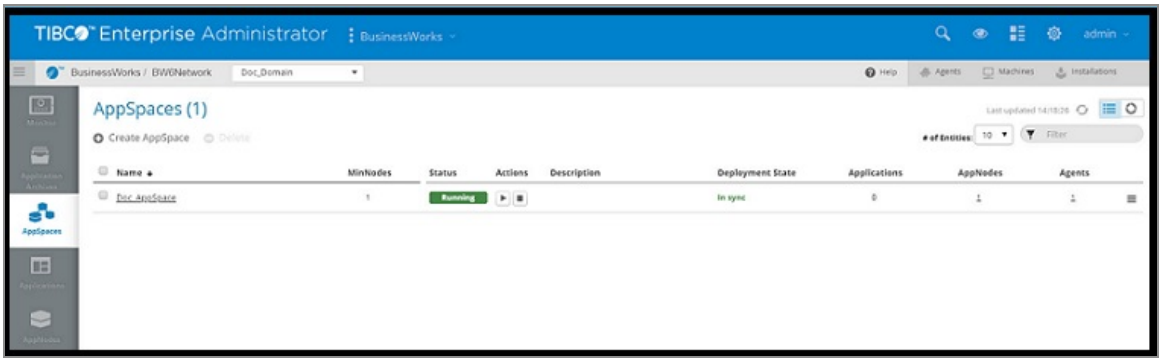
9. Add an AppNode to the AppSpace.
    a. Click the **AppNodes** icon to open the AppNodes page.
    b. Click **Create AppNode**.
    c. In the Create AppNode dialog, enter the name of the AppNode in the **Name** field.
    d. Select the agent registered with the TEA server from the **Agent** dropdown.
    e. Enter a value in the **HTTP Port** field (Example: 8075). This port must be available; each AppNode on the machine must be assigned to a unique HTTP management port.
    f. Leave the OSGi fields empty. (These are optional fields for debugging the AppNode. For more information, see Enabling the OSGi Console for an AppNode.)
    g. Select the AppSpace from the **AppSpace** dropdown.
    h. Click **Create**.

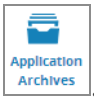
    The AppNode is created. A success notification is displayed at the top of the page and the AppNode is displayed. The AppNodes page looks similar to the following image:



10. Open the AppSpaces page and notice that the status has been updated to Stopped.
    a. Start the AppSpace by clicking the **Start** icon.

    The AppSpace starts and starts the AppNode. The status changes to Starting, then Running to indicate that both the AppNode and AppSpace are running.



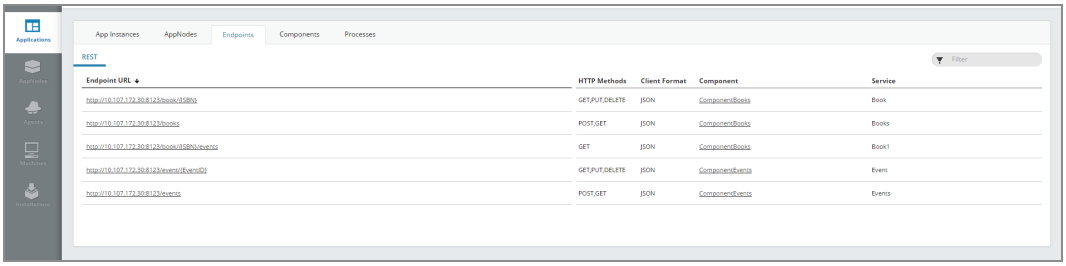
11. Upload an application archive to the AppSpace.
    a. Click the **Application Archives** icon 

.
    b. In the Application Archives page, click the **Upload** link and drag the BookStore sample application archive from *BW_HOME*`\samples\AppSpace\core\admin\ears\bookstore\ears` to the Upload Ear File dialog.
    c. Click **Upload**.
    d. A success message is displayed in the dialog. Click **Done** to close the dialog.

    The application archive is displayed on the Application Archives page:

12. Deploy the application.
    a. Click the application archive link on the Application Archives page to pivot to the application view, then click **Deploy** to deploy the BookStore application.
    b. In the Deploy Applications dialog, make sure that UI-AppSpace1 is selected from the **AppSpace** dropdown.
    c. Select **WindowsProfile.substvar (Default)** from the **Profile** dropdown.
    d. Check the **Start applications on AppNodes after deployment** option. This option starts the application after successful deployment.
    e. Click **Deploy**.

    The application is deployed to the selected AppSpace and is started on the AppNode.
13. Click the **REST Doc URL** link to view the application's REST API. You can also open the **Endpoints**, **Components**, or **Processes** tabs to drill down into the running application.



Result

You registered the BWAgent with the TEA server. You used the Admin UI to create a domain, AppSpace, and AppNode. Start the AppSpace, upload an application archive, deploy the application, and start the application. Spend some time experimenting with the Admin UI. The web interface enables you to drill down into entities and pivot views.

For more information, see Using the Admin UI.

Core Admin Sample Scripts

The sample scripts provide a simple and fast way to run the core Admin samples.

These are bash scripts. On Windows, install Cygwin64.

Admin scripts are in the following folders: *$BW_HOME*/samples/core/admin and *$BW_HOME*/scripts/admin

For information about running the sample scripts, see the "Running Admin Sample Scripts" section in *TIBCO ActiveMatrix BusinessWorks™ Getting Started*.

This sets the *TIBCO_HOME*, *BW_HOME*, *TEA_HOME*, *EMS_HOME*, and *JAVA_HOME* environment variables necessary to run the admin scripts.

All scripts support the -h and command-line argument with supported documentation for each script.

Location of the Admin Scripts

The admin scripts are located in the following folders:

- Sample scripts: $BW_HOME/samples/core/admin
- Generic scripts for ActiveMatrix BusinessWorks: $BW_HOME/scripts/admin

The scripts are updated to rely on the PATH setting to find the generic scripts. To make this easier to configure, after installation you can generate $BW_HOME/scripts/bashrc.sh that can be sourced from your ~/.bashrc.

Source the $BW_HOME/scripts/bashrc.sh to set up the following environment variables required to run the scripts mentioned in the table below:

Variable	Required
TIBCO_HOME	Yes
BW_HOME	Yes
TEA_HOME	Not required unless you run TIBCO® Enterprise Administrator on this machine.

Variable	Required
EMS_HOME	Not required unless you configure TIBCO Enterprise Message Service™ on this machine.
PATH	This variable is auto-populated based on the values set for the above variables.

Core Admin Scripts

The following table lists some of the available scripts; browse the folder for the complete list.

Core Admin Scripts

Script	Description	Location
AppManage.sh	This is an ActiveMatrix BusinessWorks 6.x utility program that emulates ActiveMatrix BusinessWorks 5.x AppManage commands. The main purpose of this utility is to demonstrate how the AppManage commands from ActiveMatrix BusinessWorks 5.x translate to the corresponding TIBCO ActiveMatrix BusinessWorks 6.x BWAdmin commands. This utility creates a cmd/AppManage_deploy.cmd folder that contains BWAdmin commands and uses bwadmin -f cmd/AppManage_deploy.cmd to run it. **Note:** Not all AppManage commands are implemented in this emulation utility. **ActiveMatrix BusinessWorks Augmented Options**: • -appSpace or - a - AppSpace name to be used for Application lifecycle.	*$BW_HOME* /samples/core/admin

Script	Description	Location
	• `-profile` or `-p` - Configuration Profile to use for deployment. This profile must be available in the EAR file. • `-profileFile`- Configuration Profile file to use for deployment. • `-debug` - Turn on debug tracing for this utility. • `-sapp` - Single Application per AppSpace deployment mode. Each AppSpace supports only one application deployment. • `-mapp` - Multiple Applications per AppSpace deployment mode. Each AppSpace supports one or more application deployments. **Note:** ActiveMatrix BusinessWorks supports both `-sapp` and `-mapp` modes. The default is `-mapp` mode.	
`bootstrap.sh`	Usage: `bootstrap.sh [-h|-help] [-clean] [-forceClean|-force|-forceclean]` This utility is a wrapper script around the following scripts: • `killtea.sh` • `killbwagent.sh` • `teaclean.sh` only if `-clean` or `-forceClean` options is used. • `bwclean.sh` if and only if `-clean` or `-forceClean` options is used. • `genbwagentini.sh`	*`$BW_HOME`*`/scripts/admin`

Script	Description	Location

	• `tea.sh`
	• `bwagent.sh`
	• `registeragent.sh`
	
	`[-h]` or `[-help]` - Prints this usage message.
	
	`-clean` Cleans TIBCO Enterprise Administrator Server Data Store and ActiveMatrix BusinessWorks Domain Data Store.
	
	Note: The `-clean` command on the data store is not reversible, so back up your data stores before using the command. Use this option carefully, as you may lose all your configurations if you do not have a backup.
	
	`-forceClean` Same as `-clean`, except it avoids prompting the user to confirm with clean.
	
	`-force` Same as `-forceClean`
	
	`-forceclean` Same as `-forceClean`
	
	This script assumes that the following products are installed correctly and the environment variables are set accordingly:
	
	TIBCO_HOME = *TIBCO_HOME* directory where you installed ActiveMatrix BusinessWorks.
	
	TEA_HOME = Parent directory to TIBCO Enterprise Administrator's `/bin` directory.
	
	Supports generation of `bwagent.ini` file for either Database/ TIBCO EMS™, or	

Script	Description	Location
	Database/ TIBCO FTL® as the technology type.	
`bounce.sh`	This utility performs these actions: 1. Stops TIBCO Enterprise Administrator Server and BWAgent Processes. 2. Restarts TIBCO Enterprise Administrator Server and BWAgent Processes. 3. Registers BWAgent to TIBCO Enterprise Administrator Server. `[-h]` or `[-help]` - Prints this help message and exits.	*$BW_HOME* `/scripts/admin`
`bounceagent.sh`	Ends and restarts BWAgent Process. `[-h]` or `[-help]` - Prints this help message and exits.	*$BW_HOME* `/scripts/admin`
`bwadmin.sh`	This is a utility script that wraps around the `bwadmin` executable. `[-h]` or `[-help]` - Prints this help message and exits. `[-network <bwagent Network Name>]` - Connects to a named BWAgent Network. This is an optional argument. By default, this script uses `$BW_HOME/config/bwagent.ini` `[<bwadminArgs> ...]` - Use BWAdmin to run commands found in the input files. Start BWAdmin in the interactive mode if cmdFile is not specified.	*$BW_HOME* `/scripts/admin`

Script	Description	Location
	A BWAgent Network Name is a named directory under `${TIBCO_HOME}/bw/networks` and contains the corresponding `bwagent.ini`. **Setting up a Newly Named Network** 1. Obtain a `bwagent.ini` created for the named BWAgent network. For example, a named network called `acmeNetwork`. 2. Create the `acmeNetwork` directory under `${TIBCO_HOME}/bw/networks`. For example, `mkdir ${TIBCO_HOME}/bw/networks/acmeNetwork` 3. Copy `bwagent.ini` to the above directory. 4. Rerun `bwadmin.sh -network acmeNetwork`	
`bwagent.sh`	This script starts BWAgent in the background and waits until it is fully initialized, or the maxWait time (<n> * 2 sec) expires. `[-h]` or `[-help]` - Prints this usage message. `[-network <Network>]` - Starts up BWAgent using the configuration of a named network. `[-maxWait <n>]` - Maximum amount of wait time (2 sec increment) for BWAgent startup success. The default value for <n> is 30, which means 30 * 2 sec = 60 seconds	*$BW_HOME* `/scripts/admin`

Script	Description	Location
bwclean.sh	This utility script cleans up ActiveMatrix BusinessWorks Domain Data and internal Data Store. The end effect of this clean up is similar to a fresh installation of ActiveMatrix BusinessWorks. [-force] or [-forceClean] - Proceeds with wiping ActiveMatrix BusinessWorks Domain Data and internal Data Store without prompting user reconfirmation. By default, the script prompts user confirmation.	*$BW_HOME*/scripts/admin
configureBWEngineGroup.sh	This utility configures AppNodes in a Domain/AppSpace to form a fault-tolerant group and cross engine persistence [-h] or [-help] - Prints this usage message. [-c] or [-clean] - Cleans up and drops all the previously configured database tables. Use this option carefully. This operation cannot be undone. Do not specify both –setup and –cleanup on the same run. [-s] or [-setup] - Does the one time setup of the BWEngine Database. When this option is used, –domain and –appspace arguments are not needed and are not used even if specified. ${BW_HOME}/config/sqlscripts/<dbtype>/create.sql is used to set up the database tables and configuration. [-b] or [-bootstrap] - Does clean up then setup. [-t] or [-dbtype] - This is the default	*$BW_HOME*/scripts/admin

Script	Description	Location
	value is `postgresql`. `-cf <config.sh>` - Sources configuration from the specified `<config.sh>` file. By default, `<$BW_HOME>/scripts/admin/config/bwengine-group-<dbtype>.sh` `[-d]` or `[-domain]` - Domain Name `[-a]` or `[-appspace <appspace>]` - AppSpace Name. All AppNodes in the specified Domain and AppSpace are configured to form a Fault-Tolerant group and across engine persistence.	
`deploy.sh`	Usage: `deploy.sh -ear <EARFile> [-h|-help] [-domain <DomainName>] [-appspace <AppSpaceName>] [-redeploy|-force] [-profile <Profile>]` Deploys the specified ActiveMatrix BusinessWorks EAR File into `-domain <DomainName> -appspace <AppSpaceName>` `[-h]` or `[-help]`- Prints this help message. `-ear <EARFile>` - Enterprise Archive file to deploy `[-domain <DomainName>]` - Domain Name - Optional parameter If not specified, DomainName is computed from `${USER}-Domain` This utility creates the Domain if it does not exist. `[-appspace <AppSpaceName>]` - AppSpace Name - Optional parameter	*$BW_HOME*`/scripts/admin`

Script	Description	Location
	If not specified, AppSpaceName is computed from the name of the EAR file. This utility creates the AppSpace and AppNode if they do not exist. `[-redeploy|-force]` - Redeploy if the application has been previously deployed. The application is not redeployed if it exists and this option is not specified. `[-profile <Profile>]`: Profile name to use for this deployment. If not specified, the default profile as packaged in the Enterprise Archive file is used. `[-mapp]` - Optional flag to set Multiple Applications per AppSpace Mode. This is the default mode for ActiveMatrix BusinessWorks. `[-debug]` - Prints debug tracing for this script `./deploy.sh`	
`genbwagentini.sh`	This script auto generates `${BW_HOME}/config/bwagent.ini` based on configurations defined in `./config/bwadmin-default- config.sh` `-h` or `-help` - Prints this help message. The following variables are required from `./config/bwadmin-default- config.sh`: • `BWAgentNetworkName` - Name of BWAgent Network. • `BWMachines` - Defined as a list of machine names (as obtained	*$BW_HOME* `/scripts/admin`

Script	Description	Location
	through `hostname -f`). If you have only one machine to configure, do not add it to this list because this script auto-configures it as a standalone BWAgent Network. This script uses `hostname -f` to determine the name of the machine it is run on. It then determines whether this machine is in the `BWMachines` list. You can assume that the `discoveryURL` of the `bwagent.ini` is comparable to that of a Database Server's URL, and the `BWAgentNetworkName` is then comparable to the Database Name. You can configure both to access the specific instance of the Database uniquely. If the `KEEP_BWAGENT_INI` environment variable is defined, the `bwagent.ini` generation is skipped. You can edit either the `./config/bwadmin-default- config.sh` file, or make a copy of it, edit it, and then set the environment variable `BWADMIN_CONFIG` to point to it. For example, export `BWADMIN_CONFIG=~/config/bwadmin-my-config.sh` Generates the `bwagent.ini` file for either Database/EMS, or Database/ TIBCO FTL® as the technology type.	
`kill.sh`	Ends all processes that match the specified <*process name*> `-h` or `-help` - Prints this help message. <*process name*> - Name of the process you	*`$BW_HOME`*`/scripts/admin`

Script	Description	Location
	want to end. This script ends all instances of the `pid` that matches this name.	
`killall.sh`	This script finds and ends all instances of processes that match the following names: • `tea` • `bwagent` • `bwappnode` • `bwadmin` `-h` or `-help` - Prints this help message.	*`$BW_HOME`*`/scripts/admin`
`killbwagent.sh`	This script finds and ends all instances of processes that match `bwagent`. `-h` or `-help` - Prints this help message.	*`$BW_HOME`*`/scripts/admin`
`killbwappnodes.sh`	This script finds and ends all instances of processes that match `bwappnode`. `-h` or `-help` - Prints this help message.	*`$BW_HOME`*`/scripts/admin`
`killtea.sh`	This script finds and ends all instance of processes that matches `tea`. `-h` or `-help` - Prints this help message.	*`$BW_HOME`*`/scripts/admin`
`killtibemsd64.sh`	This script finds and ends all instances of processes that match `tibemsd`. `-h` or `-help` - Prints this help message.	*`$BW_HOME`*`/scripts/admin`
`recreatedb.sh`	This script cleans up and recreates the PostgreSQL DB needed by the ActiveMatrix BusinessWorks BookStore REST sample in: `${BW_HOME}/samples/binding/rest/BookStore`	*`$BW_HOME`*`/samples/core/admin`

Script	Description	Location
	`-h` and `-help` - Prints this help message.	
`registeragent.sh`	This utility registers the local BWAgent with the TIBCO Enterprise Administrator server. `-h` or `-help` - Prints this help message. This utility assumes that the following environment variables have been set: Export TIBCO_HOME="<Where ActiveMatrix BusinessWorks is installed>" At least one of the following environment variables is set: Export TEA_HOME="Where TIBCO Enterprise Administrator is installed in the form of $TIBCO_HOME/tea/<version>" Or, Export TEA_HOSTNAME=<HostName> If the *TEA_HOSTNAME* environment variable is set, it assumes that the TIBCO Enterprise Administrator server is running remotely from the local BWAgent instance. If the *TEA_HOSTNAME* environment variable is not set, this script registers the local BWAgent to the locally running TIBCO Enterprise Administrator server.	*$BW_HOME*`/scripts/admin`
`runAcme.sh`	Creates <domain> and deploys all EAR files found under `${BW_HOME}/samples/core/admin/ears/acme`. `-h` or `-help`: Displays this usage message `<domain>` - can be "Acme-QA-Domain" or	*$BW_HOME*`/samples/core/admin`

Script	Description	Location
	"Acme-UAT-Domain". When not specified, the default is "Acme-QA-Domain" `<mode>` - `[-sapp ]` or `[-mapp]` `-sapp` - Single App AppSpace deployment mode. Each AppSpace supports only one application deployment. `-mapp` - Multiple App AppSpace deployment mode. Each AppSpace supports one or more application deployments. **Note:** ActiveMatrix BusinessWorks supports both `-sapp` and `-mapp` modes. The default is `-mapp` mode. This script dynamically creates a `bwadmin` command file in `cmd/<domain>-<mode>.cmd` and runs it.	
`runAll.sh`	This utility is a wrapper script that performs the following: • bootstrap.sh - only if running in a single machine setup • runBookStore.sh • runSamples.sh • runAcme.sh -domain Acme-QA-Domain • runAcme.sh -domain Acme-UAT-Domain `-h` or `-help` - Displays this usage message and exits `-clean` - Cleans the TIBCO Enterprise	*`$BW_HOME`* `/samples/core/admin`

Script	Description	Location
	Administrator Server Data Store and ActiveMatrix BusinessWorks Domain Data Store. The data store clean is not reversible. Make sure you back up your data stores before running this command. Use this option with utmost care, otherwise you risk losing all your configurations. `-forceClean` - Same as `-clean`, except it avoids prompting you to confirm with clean. `-force` - Same as `-forceClean` `<mode>` - [`-sapp` | `-mapp`] `-sapp` - Single Application per AppSpace deployment mode. Each AppSpace supports only one application deployment. `-mapp` - Multiple Applications per AppSpace deployment mode. Each AppSpace supports one or more application deployments. **Note:** ActiveMatrix BusinessWorks supports both `-sapp` and `-mapp` modes. The default is `-mapp` mode. Generates the `bwagent.ini` file for either Database/EMS™, or Database/FTL® technology type.	
`runBookStore.sh`	Creates BookStore-Domain and deploys all EAR files found under `${BW_HOME}/samples/core/admin/ears/bookstore` `-h` or `-help` - Displays this usage message.	*`$BW_HOME`*`/samples/core/admin`

Script	Description	Location
	<mode> - [-sapp | -mapp] -sapp - Single Application per AppSpace deployment mode. Each AppSpace supports only one application deployment. -mapp - Multiple Application per AppSpace deployment mode. Each AppSpace supports one or more application deployments. ActiveMatrix BusinessWorks supports both -sapp and -mapp modes. The default is -mapp mode. **Note:** This script dynamically creates a bwadmin cmd file in cmd/Samples-Domain-<mode>.cmd and runs it.	
runSamples.sh	Creates Samples-Domain and deploys all EAR files found under ${BW_HOME}/samples/core/admin/ears/samples -h or -help - Displays this usage message. <mode>: [-sapp] or [-mapp] -sapp: Single Application per AppSpace deployment mode. Each AppSpace supports only one application deployment. -mapp: Multiple Application per AppSpace deployment mode. Each AppSpace supports one or more application deployments.	*$BW_HOME*/samples/core/admin

Script	Description	Location
	Note: ActiveMatrix BusinessWorks supports both `-sapp` and `-mapp` modes. The default is `-mapp` mode. This script dynamically creates a `bwadmin` command file in `cmd/Samples-Domain-<mode>.cmd` and runs it.	
`showprocs.sh`	Shows process ID and complete binary path of all processes required in ActiveMatrix BusinessWorks: • `tibemsd` • `tea` • `bwagent` • `bwappnode` • `bwadmin`	*$BW_HOME* `/scripts/admin`
`tea.sh`	This script starts TIBCO Enterprise Administrator in the background and waits until it is completely initialized, or the maxWait time (<n> * 2 sec) expires. `-h` or `-help` - Prints this usage message. [-maxWait <n>] - Maximum wait duration (2-second increment) before TIBCO Enterprise Administrator Server starts successfully. The default value for <n> is 30, which means 30 * 2 sec = 60 seconds.	*$BW_HOME* `/scripts/admin`
`teaclean.sh`	This utility script cleans the TIBCO Enterprise Administrator Server's configuration data store. The result of this clean up is similar to a	*$BW_HOME* `/scripts/admin`

Script	Description	Location
	fresh installation of TIBCO Enterprise Administrator. `-h` or `-help` - Prints this usage message `-force` or `-forceClean` - Proceeds with wiping ActiveMatrix BusinessWorks Domain Data and internal data store without prompting user reconfirmation. By default, the script prompts user confirmation.	
`tibemsd64.sh`	This script starts `tibemsd64` in the background and waits until it is completely initialized, or the maxWait time (<n> * 2 sec) expires. `-h` or `-help` - Prints this usage message `[-maxWait <n>]` - Max number of wait time (2-seconds increment) for `tibemsd64` startup success. The default value for *<n>* is 30, which means 30 * 2 sec = 60 seconds. This script is only supported on UNIX-based systems. For Windows, use Windows Systems Services to start or stop `tibemsd64`.	*$BW_HOME* `/scripts/admin`

Note: Each `runAcme.sh`, `runBookStore.sh`, `runSamples.sh`, `deploy.sh`, and `AppManage.sh` generates `bwadmin` commands before execution.

The generated `bwadmin` command files are found under the `cmd` subdirectory.

Administrator and Agent

BWAdmin and BWAgent are used to create, manage, and monitor domains, AppSpaces, AppNodes, archives, and applications.

For more information, see BWAdmin and BWAgent.

Runtime entities are created in the local file system in the *BW_HOME*/domains folder. The default location of this folder can be changed. For information, see Configuring the Location of the Domains Folder.

BWAdmin

BWAdmin provides a command-line console that can be used in local mode or enterprise mode to create and manage domains, AppSpaces, AppNodes, archives, and applications. Collectively, the entities provide the logical and physical structure for the runtime environment.

The BWAdmin provides the following features:

- One tool for both local and enterprise mode with identical commands
- Interactive shell
- Batch/silent mode by passing a command file as an argument
- Ability to run commands locally as well as remotely
- Ability to address different BWAgent networks
- Simple and intuitive command structures
- Nested commands
- Unix-style commands for complex scripting
- Command completion

A full range of commands is available. Command can be run stand-alone from the command line or from the BWAdmin console. Unix-style scripts can be created to run

BWAdmin commands. When scripting, you may need to include conditions for possible error codes.

For more information about error codes and the corrective action to take, see the *TIBCO ActiveMatrix BusinessWorks™ Error Codes* guide.

Commands can be issued from the following modes:

- Interactive Mode: Useful for exploration. Commands are run from the BWAdmin shell. Any number of commands can be run in a sequence.
- Command Line: Useful for execution of single commands. Commands are run stand-alone from the command line with the provided syntax.
- Batch Mode: Useful for execution of repetitive commands.

To get help on a command, including syntax information, type `help` followed by the command name, from either interactive mode or the command line. Example:

```
bwadmin help
bwadmin help create
bwadmin help registerteaagent
```

Interactive Mode

Interactive mode is used for exploring runtime entities. Enter interactive mode by typing `bwadmin` at the command line. To view a list of available commands, press `tab`.

The `cd` command sets the runtime entity context so you can omit runtime entity options for commands like `create`, `delete`, `start`, or `stop`.

Command Line

The BWAdmin commands can be issued from the command line in the format: `bwadmin [options] command <arguments>`

To see the list of all BWAdmin commands, type `bwadmin help` at the command line.

The following options can be specified for BWAdmin at the command line:

BWAdmin Command Options

Option	Description	Example
`-b/--batch`	Reads a series of commands from the standard input.	`bwadmin --batch | bwadmin get admin.mode`
`-config`	Applies the configuration in the specified file to the server instance.	`bwadmin -config -d myDomain -a myAppSpace -cf` *file_path*`/config.ini`
`-D <property=value>`	Applies the specified value to the specified property. Use the BWAdmin `get` command to retrieve the value.	`bwadmin -D name=User1`
`-f <file[,<file>]`	Reads commands from the specified file or from the comma-separated list of files. The specified file can contain one command or multiple commands. Exits after command execution is completed.	`bwadmin -f backupMyAppNode.cmd`
`-l/--login <arg>`	Specifies the log in ID to use for the session.	Given the following command: `bwadmin -l User1` BWAdmin in interactive mode displays: `bwadmin[User1]>`
`-logconfig <file>`	Uses the specified file for the Logback configuration.	`bwadmin -logconfig mylogback.xml` For more information about logging, see Logging.
`-x,--xtrace`	Echoes the command to the terminal.	Given the following command `bwadmin -x create domain`

Option	Description	Example

		`MyDomain1` the following sample output is issued:
		`TIBCO ActiveMatrix BusinessWorks version 6.2.0, build V20, 2014-10-09`
		`+ create domain MyDomain1`

`exit`	Exits the command-line console.	`bwadmin>exit` exits the command-line console.

For more information about BWAdmin commands for different administration tasks, see the "BWAdmin Command Line" topics in the Administration Tasks and Reference section.

Batch Mode

A command file can be passed to BWAdmin at the command line with the `-f` option. The batch file should contain all required inputs. An example of a command file is a backup file created with the `backup` command.

The `-f eoe` command is an optional command and you can run the BWAdmin commands in a batch mode. If any of the commands fail, the subsequent commands are not run. Syntax for the command is `bwadmin.exe -f eoe <command file>`.

For example: `bwadmin.exe -f eoe bwadmin.sh`

BWAgent

A BWAgent is a daemon process that is responsible for provisioning AppNodes and applications, performing administration commands, and synchronizing data from the datastore with the local file system.

There is one BWAgent for each installation. The BWAgent enables communication between agents on different machines. When multiple BWAgents are configured to communicate with each other using a common datastore, they form a BWAgent network. Bwagents can communicate using TIBCO FTL® for communication transport and TIBCO Enterprise Message Service for communication transport, and by using an external database for data persistence.

For information about configuring the BWAgent, see Configuring BWAgent.

When multiple BWAgents belong to a network and one of the systems fails, the failed system can be restored after a restart by using the BWAdmin `restore` command to force the file system to be synchronized with the datastore.

There are multiple ways to access the BWAgent: BWAdmin, the Admin UI, or the REST API.

- BWAdmin: In enterprise mode, the BWAdmin sends commands to the BWAgent. The BWAgent dispatches the command to the targeted agent. For more information, see the "BWAdmin Command Line" tasks under Administration Tasks and Reference section in the *TIBCO ActiveMatrix BusinessWorks™ Administration* guide.
- Admin UI: When the BWAgent is registered with the TEA server, the Admin UI can be used to create and manage runtime entities. For more information, see the "Admin UI" tasks under "Administration Tasks and Reference" section in the *TIBCO ActiveMatrix BusinessWorks™ Administration* guide.
- REST API: View the BWAgent REST API in the Swagger UI. For more information, see the section "Accessing the BWAgent REST API with the Swagger UI" in the *TIBCO ActiveMatrix BusinessWorks™ Administration* guide.

The BWAgent supports its own set of commands. Commands are issued from the command line in the format:

```
bwagent [options] command <arguments>
```

BWAgent commands are listed below.

BWAgent Commands

Command	Description
`apiserver`	Starts the apiserver that hosts the REST API in the Swagger UI. Open a browser and go to the following URL: `http://localhost:5555`

Command	Description
`startagent`	Starts the BWAgent. This is the same as the default command when no command is given.
`stop`	Stops the BWAgent gracefully.

The following options can be specified for the BWAgent:

BWAgent Command Options

Option	Description	Example
`-config`	Applies the configuration in the specified file to the server instance.	`bwagent -config bwagent.ini`
`-logconfig <file>`	Uses the specified file for the Logback configuration.	`bwagent -logconfig mylogback.xml`
`-x,--xtrace`	Echoes the command to the terminal.	Given `bwagent -x` , the text `+startagent` is echoed to the console when the agent starts.

Configuring BWAgent

The BWAgent can be configured for a multi-agent, multi-machine environment.

The BWAgent is configured using the `bwagent.ini` file in the *`BW_HOME`*`\config` folder. The `bwagent.ini` file template is a configuration file and contains the BWAdmin datastore configuration properties. Properties are pushed to the configuration file using JSON files.

BWAgent Configuration Properties

Properties	Description
BWAgent general configuration	

Properties	Description
`bw.admin.mode`	The Admin mode. BW Administration tools can work in two modes, enterprise mode or local mode. In the enterprise mode, it works with the agents across machines. In the local mode, it works only with a local machine and assumes no data store and transport and agents are available. The actions performed in local mode are not visible to the agents whenever they are started, or even the admin tool is started in the enterprise mode.
`bw.agent.networkName`	The name of the network. Must be the same for all the BWAgents in the network. For more information, see Creating an Agent Network.
`bw.agent.memberName`	The name of the BWAgent. Must be unique within the network. For more information, see Creating an Agent Network.
`bw.agent.technology.db.create.schema`	Add this property manually to the `bwagent.ini` file. Set the property to true to allow BWAgent to run create table script on startup. When you set the property to false, the default behavior of BWAgent changes, and it restricts BWAgent from running the create table script on startup.
Logging configuration	
`logback.configurationFile`	The Logback configuration file to be used by the agent.
`bw.agent.http.port`	The HTTP port.
`bw.agent.http.host`	The HTTP interface (default=localhost).

Properties	Description
bw.appnode.agent.http.communication.port	The internal HTTP communication port that the Thor engine uses to communicate with the BWAgent to send the status of AppNodes and applications. Update this property to specify a port to start the internal server on. The default port number is 56565.
bw.agent.http.access.log.config	The HTTP Request Access Log Configuration file.
bw.agent.bw.auth	The authentication mechanism used by REST API, BASIC (default), or DIGEST.
bw.agent.https.port	The secure port.
bw.agent.https.truststorepath	The truststore.
bw.agent.https.truststorepassword	The truststore password
bw.agent.https.keystorepath	The keystore.
bw.agent.https.keystorepassword	The keystore password.
bw.agent.https.excludeprotocols	The protocols to be excluded.
bw.agent.https.includeprotocols	The protocols to be included.
Configuration for AppNode to agent communication	
bw.agent.appnode.user	The user used by the AppNodes to communicate with the BWAgent.
bw.agent.appnode.password	The password for the user used by AppNodes to communicate with the BWAgent. If not set, the obfuscated password is read from the configured realm file. For example, $BW_HOME/config/realm.properties.

Properties	Description
`bw.agent.appnode.status.notify.timeout`	Time interval in seconds when the AppNode reports its status to the BWAgent.
TEA Agent configuration	
`bw.agent.tea.agent.host`	Identifies the BWAgent for TEA to be registered.
`bw.agent.tea.agent.port`	Identifies the BWAgent for TEA to be registered.
`bw.agent.tea.agent.context.path=/bwta`	Used to create BWAgents register URL for TEA.
`bw.agent.tea.server.url`	The BWAgent uses this URL to identify which server to be registered to.
Technology Type Configuration. Supported types are DBEMS or DBFTL	
`bw.agent.technology.type`	The provider to use for the datastore such as an external database (PostgreSQL, MySQL, Microsoft SQL, Oracle, db2, or MariaDB) with transport as TIBCO FTL® or TIBCO Enterprise Message Service (EMS). Set to either: • DBFTL • DBEMS For more information, see the following topics: • Database with FTL Configuration for BWAgent • Database with EMS Configuration for BWAgent
`bw.agent.technology.requestTimeout`	Timeout for requests sent to other BWAgents. The default value is 60000 ms.

Properties	Description
DBEMS technology type	
bw.agent.technology.dbems.db.provider	Database provider. Supported options are PostgreSQL, MySQL and Oracle database 12c, MS sqlserver, db2, and MariaDB.
bw.agent.technology.dbems.db.driver	The DB driver. Examples: • dbDriver=org.postgresql.Driver • dbDriver=com.mysql.jdbc.Driver • dbDriver=oracle.jdbc.OracleDriver • dbDriver=com.microsoft.sqlserver.jdbc.SQLServerDriver • dbDriver=com.ibm.db2.jcc.DB2Driver • dbDriver=org.mariadb.jdbc.Driver
bw.agent.technology.dbems.db.connectionURL	The DB connection URL. Examples: • dbConnectionURL=jdbc:postgresql://db:5432/bwadmindb • dbConnectionURL=jdbc:mysql://db:3306/bwadmindb • dbConnectionURL=jdbc:oracle:thin:@db:1521:bwadmindb • dbConnectionURL=jdbc:sqlserver://db:1433;databaseName=bwadmindb • dbConnectionURL=jdbc:db2://db:50000/bwadmindb • dbConnectionURL=jdbc:mariadb://db:330

Properties	Description
	`6/databaseName=bwadmindb`
`bw.agent.technology.dbems.db.userName`	The DB user. Example: `dbUserName=bwuser`
`bw.agent.technology.dbems.db.password`	The DB password. Example: `dbPassword=bwuser`
`bw.agent.technology.dbems.ems.serverUrl`	The EMS server URL. Example: `emsServerUrl=tcp://ems:7222` Example: `ldap://nn.nn.nnn.nnn:nnnnn/CN=admin,ou=users,o=tibco` **Note:** Provide a comma-separated list to add multiple EMS servers.
`bw.agent.technology.dbems.ems.connectionFactoryName`	The EMS connection Factory name. Example: `ldap.connectionFactoryName=QCFN`
`bw.agent.technology.dbems.ems.userName`	The EMS user. Example: `emsUserName=admin`
`bw.agent.technology.dbems.ems.password`	The EMS user password. Example: `emsPassword=`
`bw.agent.technology.dbems.ems.requestQueueName`	The EMS member queue. Example: `requestQueueName=bw6.admin.operations.queue.{{membername}}`
`bw.agent.technology.dbems.ems.qin.EMSPrefix`	The BWAgent Qin group name prefix. This property is optional and the default value is

Properties	Description
	"EMSGMS".
bw.agent.technology.dbems.ems.ssl.trust.identity	The EMS SSL configuration. Client identity consisting of the certificate, private key, and optionally extra issuer certificates can be included into a single data block using PKCS12, Keystore, or Entrust Store encodings. Example:bw.agent.technology.dbems.ems.ssl.trust.identity={EMS_HOME}/samples/certs/client_identity.p12
bw.agent.technology.dbems.ems.ssl.trust.cert.location	The set of trusted certificates represents all trusted issuers of the server certificate. It must be specified by the client application unless the host certificate verification is completely disabled. Example: bw.agent.technology.dbems.ems.ssl.trust.location={EMS_HOME}/samples/certs/server_root.cert.pem
bw.agent.technology.dbems.ems.ssl.trust.password	EMS SSL connection trust password. This property is required if the JMS server protocol is SSL. The password may be clear text or supplied as an obfuscated string.
bw.agent.technology.dbems.ems.ssl.disable.verify.host.name	The trusted certificate common name must match the EMS server hostname if set to false.
bw.agent.technology.dbems.ems.ssl.disable.verify.host	The client and server certificates must match if set to false.
bw.agent.technology.dbems.ems.reconnection.interval	Interval for EMS reconnection. Value is in milliseconds (default: 10 milliseconds).

Properties	Description
DBFTL technology type	
bw.agent.technology.dbftl.db.provider	The Database provider. Supported options are PostgreSQL, MySQL and Oracle database 12c, MS sqlserver, db2, and MariaDB.
bw.agent.technology.dbftl.db.driver	The DB driver. Examples: • dbDriver=org.postgresql.Driver • dbDriver=com.mysql.jdbc.Driver • dbDriver=oracle.jdbc.OracleDriver • dbDriver=com.microsoft.sqlserver.jdbc.SQLServerDriver • dbDriver=com.ibm.db2.jcc.DB2Driver • dbDriver=org.mariadb.jdbc.Driver
bw.agent.technology.dbftl.db.connectionURL	The DB connection URL. Examples: • dbConnectionURL=jdbc:postgresql://db:5432/bwadmindb • dbConnectionURL=jdbc:mysql://db:3306/bwadmindb • dbConnectionURL=jdbc:oracle:thin:@db:1521:bwadmindb • dbConnectionURL=jdbc:sqlserver://db:1433;databaseName=bwadmindb • dbConnectionURL=jdbc:db2://db:50000/bwadmindb •

Properties	Description
	`dbConnectionURL=jdbc:mariadb://db:3306/databaseName=bwadmindb`
`bw.agent.technology.dbftl.db.userName`	The DB user. Example: `dbUserName=bwuser`
`bw.agent.technology.dbftl.db.password`	The DB password. Example: `dbPassword=bwuser`
`bw.agent.technology.dbftl.ftl.realmserver`	The FTL Realm server URL. Example: `ftlRealmServerUrl=http://localhost:8070`
`bw.agent.technology.dbftl.ftl.application`	The FTL application name. Example: `ftlApplicationName=bwadmin`
`bw.agent.technology.dbftl.ftl.identifier`	The FTL identifier. Example: `ftlIdentifier=`
`bw.agent.technology.dbftl.ftl.secondary`	The FTL secondary realm server. Example: `ftlSecondaryUrl=http://localhost:8070`
`bw.agent.technology.dbftl.ftl.username`	The FTL user. Example: `ftlUserName=admin`
`bw.agent.technology.dbftl.ftl.password`	The FTL user password. Example: `ftlPassword=`
`bw.agent.technology.dbftl.ftl.endpoint`	The FTL endpoint. Example: `ftlEndpoint=bw-endpoint`

Properties	Description
`bw.agent.technology.dbftl.ftl.dataformat`	The FTL data format. Example: `ftlDataformat=`
`bw.agent.technology.dbftl.ftl.inbox`	The FTL inbox name. Example: `ftlInbox=`
Statistics Provider Configuration	
`bw.agent.technology.statsProvider`	The stats provider technology.
`bw.agent.technology.statsProvider.db.provider`	The Database provider. Supported options are PostgreSQL, MySQL and Oracle database 12c, MS sqlserver, db2, and MariaDB.
`bw.agent.technology.statsProvider.db.driver`	The DB driver. Examples: • `dbDriver=org.postgresql.Driver` • `dbDriver=com.mysql.jdbc.Driver` • `dbDriver=oracle.jdbc.OracleDriver` • `dbDriver=com.microsoft.sqlserver.jdbc.SQLServerDriver` • `dbDriver=com.ibm.db2.jcc.DB2Driver` • `dbDriver=org.mariadb.jdbc.Driver`
`bw.agent.technology.statsProvider.db.connectionURL`	The DB connection URL. Examples: • `dbConnectionURL=jdbc:postgresql://db:5432/bwadmindb` • `dbConnectionURL=jdbc:mysql://db:3306/bwadmindb`

Properties	Description
	• dbConnectionURL=jdbc:oracle:thin:@db:1521:bwadmindb • dbConnectionURL=jdbc:sqlserver://db:1433;databaseName=bwadmindb • dbConnectionURL=jdbc:db2://db:50000/bwadmindb • dbConnectionURL=jdbc:mariadb://db:3306/databaseName=bwadmindb
bw.agent.technology.statsProvider.db.userName	The DB user. Example: dbUserName=bwuser
bw.agent.technology.statsProvider.db.password	The DB password. Example: dbPassword=bwuser
Governance and Policy Director Configuration - The properties in this section are applicable to the Governance Lifecycle Event Listener and it is used to communicate with the TIBCO Policy Director Administrator.	
bw.governance.enable	To enable or disable the Governance Lifecycle Event Listener. This property is optional and specifies whether the Governance Lifecycle Event Listener should be enabled or disabled in the BWAgent. The supported values are true or false. The default value is false.
bw.governance.jms.server.url	The Policy Director Administrator JMS URL. This property is optional and is used to specify the JMS server URL used to communicate with the Policy Director Administrator. If this property is not set, then the Lifecycle Event Listener does not attempt to connect to the JMS server. The URL is expected to start with tcp:// or ssl://

Properties	Description
	and the failover URLs can be specified as a ',' or '+' separated list.
bw.governance.jms.server.username	The Policy Director Administrator JMS Username. This property is required if the Policy Director Administrator JMS URL is specified.
bw.governance.jms.server.password	The Policy Director Administrator JMS User Password. This property is required if the Policy Director Administrator JMS URL is specified.
bw.governance.jms.ssl.trust.store.type	The Policy Director Administrator JMS SSL connection trust store type. This property is required if the JMS server protocol is SSL. The supported values are 'JKS' and 'JCEKS'. The default value is 'JKS'.
bw.governance.jms.ssl.trust.store.location	The Policy Director Administrator JMS SSL connection trust store location. This property is required if the JMS server protocol is SSL.
bw.governance.jms.ssl.trust.store.password	The Policy Director Administrator JMS SSL connection trust store password. This property is required if the JMS server protocol is SSL. The password may be clear text or supplied as an obfuscated string.
bw.governance.jms.reconnect.attempt.count	The Policy Director Administrator JMS Connection attempt count. This property is required if the Policy Director Administrator JMS URL is specified. It specifies the number of JMS connection attempts the Lifecycle Event Listener makes. The default value is '120'.
bw.governance.jms.reconnect.attempt.timeout	The Policy Director Administrator JMS Connection attempt timeout. This property is required if the Policy Director Administrator JMS

Properties	Description
	URL is specified. It specifies the timeout interval between the attempt to reestablish connection to # the JMS server. The default value is '500'.
`bw.governance.jms.reconnect.attempt.delay`	The Policy Director Administrator JMS Connection attempt delay. This property is required if the Policy Director Administrator JMS URL is specified. It specifies the delay in milliseconds between attempts to establish and reestablish the connection of the JMS server. The default value is '500'.
`bw.governance.jms.queue.pd.receiver.name`	The Policy Director Administrator JMS receiver queue name prefix. This property is required if the Policy Director Administrator JMS URL is specified. It specifies the receiver queue name prefix for the Lifecycle Event Listener and Policy Director Administrator communication. This property value must match the value specified in the Policy Director Administrator configuration. The default value is 'governance.de.bw.default'.
`bw.governance.jms.application.property.<UserCustomProperty>`	The Policy Director Administrator JMS JNDI custom property. This property is optional and it provides the ability to specify the custom property for the JMS JNDI Initial Context. For example to provide a custom property called "myProperty" for the JNDI Initial Context, then specify a property "bw.governance.jms.application.property.myProperty=".

The default location of the domains folder, where runtime entities are stored, can be changed. For information, see Configuring the Location of the Default Datastore.

Database with TIBCO FTL® for BWAgent

The BWAgent can be configured to use TIBCO FTL for transport among BWAgents. PostgreSQL, MySQL, Microsoft SQL, Oracle, and DB2 are the supported databases.

Note: Use of TIBCO FTL® with TIBCO ActiveMatrix BusinessWorks™ for configuring BWAgent and for configuring group provider for the engine does not require TIBCO FTL® licenses.

Note: Regularly back up domain data using the BWAdmin `backup` command. For more information about backing up and restoring domain data, see Backing up and Restoring a Domain.

For a multi-agent, multi-machine environment using an external database, and TIBCO FTL, modify the following properties in the `bwagent.ini` file.

BWAgent properties for Multi-Agent, Multi-Machine Environments using Database/FTL

Property	Description
`bw.agent.technology.dbftl.db.provider`	Set one of the following supported database providers: • `postgresql` • `mysql` • `mssql` • `oracle` • `db2`
`bw.agent.technology.dbftl.db.driver`	The database driver.
`bw.agent.technology.dbftl.db.connectionURL`	The URL to connect to the database.
`bw.agent.technology.dbftl.db.userName`	The username to authenticate to the database.
`bw.agent.technology.dbftl.db.password`	The password to authenticate to the database.

Property	Description
`bw.agent.technology.type`	Set dbftl as the technology type for the BWAgent to use.
`bw.agent.technology.dbftl.ftl.realmserver`	Set The FTL realm server. Example: bw.agent.technology.dbftl.ftl.realmserver=http://localhost:8070 In case of FTL 6.x server in FT mode, set multiple realmserver values separated by pipe. (|). For example: bw.agent.technology.dbftl.ftl.realmserver=http://10.97.240.76:8050 | http://10.97.240.76:8051 | http://10.97.240.76:8052
`bw.agent.technology.dbftl.ftl.application`	Set the application name. Example: bw.agent.technology.dbftl.ftl.application=bwadmin
`bw.agent.technology.dbftl.ftl.identifier`	Set the FTL identifier.
`bw.agent.technology.dbftl.ftl.secondary`	Set the secondary realm server. This property is optional for FTL 5.x. **Important:** This property is available in the `bwagent.ini` file only when you set the ftlsecondary property to true in the `bwagent_ftl.json` file. By default, the property is set to false.
`bw.agent.technology.dbftl.ftl.username`	Set the FTL username.

Property	Description
`bw.agent.technology.dbftl.ftl.password`	Set the FTL password.
`bw.agent.technology.dbftl.ftl.endpoint`	Set the FTL endpoint. Example: bw.agent.technology.dbftl.ftl.endpoint=bwadmin-endpoint
`bw.agent.technology.dbftl.ftl.dataformat`	Set the FTL data format. Example: bw.agent.technology.dbftl.ftl.dataformat=bw-format
`bw.agent.technology.dbftl.ftl.inbox`	Set the FTL inbox. Example: bw.agent.technology.dbftl.ftl.inbox=bw-inbox
`bw.agent.technology.requestTimeout`	Timeout for requests sent to other BWAgents. The default value is 6000 milliseconds.
`bw.agent.technology.remote.status.requestTimeout`	Timeout for requests sent to BWAgents to find the status of AppNodes, applications, and other BWAgents. The default value is 3000 milliseconds.

For information about setting properties, see the following topics:

- PostgreSQL - For instructions, see Configuring BWAgent for PostgreSQL and TIBCO FTL.
- MySQL - For instructions, see Configuring BWAgent for MySQL and TIBCO FTL.
- Microsoft SQL - For instructions, see Configuring BWAgent for Microsoft SQL and TIBCO FTL.
- Oracle - For instructions, see Configuring BWAgent for Oracle and TIBCO FTL.
- DB2 - For instructions, see Configuring BWAgent for DB2 and TIBCO FTL.

- MariaDB - For instructions, see Configuring BWAgent for MariaDB and TIBCO FTL.

Configuring BWAgent for PostgreSQL and TIBCO FTL®

The BWAgent can be configured to use the PostgreSQL database with TIBCO FTL for transport.

Note: Use of TIBCO FTL with ActiveMatrix BusinessWorks for configuring BWAgent and for configuring group provider for engine does not require TIBCO FTL licenses.

Note: The database name must be unique per agent network if multiple networks share the same physical database. BWAgent and BWEngine support sharing the same database, users, and schemas.

Before you begin

- Install and configure TIBCO FTL on the same machine that you have installed ActiveMatrix BusinessWorks 6.x on. For more information, see the "Setting Up TIBCO FTL® for BWAgent Transport" topic in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

  Important: For the version of TIBCO FTL that is supported with the version of ActiveMatrix BusinessWorks 6.x, see the ActiveMatrix BusinessWorks readme.

- Install PostgreSQL. The PostgreSQL driver is available by default.

Procedure

1. After installing PostgreSQL, create a database bwadmindb and the database owner, bwuser as described in the following steps:
   a. Run the following commands on the psql terminal:

```
> psql -p 5432 -c "CREATE USER bwuser WITH CREATEDB PASSWORD
```

```
'bwuser';"
> psql -p 5432 -c "CREATE DATABASE bwadmindb WITH OWNER
bwuser;"
```

b. Open the pgAdmin III utility and expand **Schemas > Tables** in the **Object Browser** to view the tables in the database.

c. To add a password for the database owner, expand **Login Roles > bwuser**, right-click, and select **Properties**. Click the **Definition** tab to create and save a password.

2. Stop the BWAgent if it is running.
3. Open the `bwagent_ftl.json` file in *BW_HOME*`\config` (Windows) or `${`*BW_HOME*`}/config` (Unix).
4. Update the following properties for your environment:

Property Name	PostgreSQL Value
`dbtype`	`postgresql`
`dbdriver`	`org.postgresql.Driver`
`dbConnectionURL`	`jdbc:postgresql://localhost:5432/bwadmindb`
`dbuser`	`bwuser`
`dbpassword`	`bwuser`

5. Run the BWAdmin `config` command with the `-cf` option to push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_ftl.json agent
```

6. Restart the BWAgent.

Configuring BWAgent for MySQL and TIBCO FTL®

The BWAgent can be configured to use MySQL Server database with TIBCO FTL for transport.

Note: Use of TIBCO FTL with ActiveMatrix BusinessWorks for configuring BWAgent and configuring group provider for BWEngine do not require TIBCO FTL licenses.

Note: The database name must be unique per agent network if multiple networks share the same physical database. BWAgent and BWEngine support sharing the same database, users, and schemas.

Before you begin

- Install and configure TIBCO FTL on the same machine that you have installed ActiveMatrix BusinessWorks 6.x. For more information, see the "Setting Up TIBCO FTL® for BWAgent Transport" topic in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

  Important: For the version of TIBCO FTL that is supported with the version of ActiveMatrix BusinessWorks 6.x, see the ActiveMatrix BusinessWorks readme.

- Download the latest MySQL server package from their official website and install MySQL. Configure the server configuration by following the prompts in the MySQL Server Configuration. Ensure that you select the following values:
  - Database: Multifunctional
  - Type of connectivity: Manual
  - Default port: 3306
- Download the latest JDBC driver and connector JAR files for MySQL to the `BW_HOME\config\drivers\shells\jdbc.mysql.runtime\runtime\plugins\com.tibco.bw.jdbc.datasourcefactory.mysql\lib` folder.
- Install the MySQL driver by running the command `bwinstall mysql-driver` from the `/bin` folder.

After installing MySQL Server, create a database bwadmindb and grant privileges to the default database owner root as described in the following steps:

Procedure

1. Create a database `bwadmindb` and grant privileges to the default database owner root as described in the following steps:
    a. Run the following command on the MySQL terminal:

       For MySQL 5.x

       ```
       mysql>create database bwadmindb;
       ```

       For MySQL 8.x

       ```
       CREATE SCHEMA `bwadmindb` DEFAULT CHARACTER SET utf8;
       ```

    b. Run the following command to view the tables included in the newly created database:

       ```
       mysql>use bwadmindb;
       ```

    c. Run the following command to grant all privileges to the root user for that database after replacing the value for the host IP address:

       ```
       mysql>GRANT ALL PRIVILEGES ON *.* TO 'root'@<host_IP> IDENTIFIED
       BY 'Tibco123'WITH GRANT OPTION;
       ```

       To grant the minimum number of permissions to the `bwuser` for that database, run the following command, where *<host_IP>* is replaced with the value for the host IP address:

       ```
       grant create,select,update,insert,delete ON *.* to
       'bwuser'@<host_IP> IDENTIFIED BY 'bwuser';
       ```

2. Stop the BWAgent if it is running.
3. Open the `bwagent_ftl.json` file in *BW_HOME*`\config` (Windows) or `${`*BW_HOME*`}/config` (Unix).
4. Update the following properties for your environment:

Property Name	MySQL Value
dbtype	mysql
dbdriver	com.mysql.jdbc.Driver
dbconnectionurl	jdbc:mysql://localhost:3306/bwadmindb?useSSL=false
dbuser	bwuser
dbpassword	bwuser

5. Run the BWAdmin `config` command with the `-cf` option push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_ftl.json agent
```

6. Restart the BWAgent.

Configuring BWAgent for Microsoft SQL Server and TIBCO FTL®

The BWAgent can be configured to use Microsoft SQL database with TIBCO FTL for transport.

Note: Use of TIBCO FTL with ActiveMatrix BusinessWorks for configuring BWAgent and for configuring group provider for engine does not require TIBCO FTL licenses.

Note: The database name must be unique per agent network if multiple networks share the same physical database. BWAgent and BWEngine support the sharing of the same database, users, and schemas.

Before you begin

- Install and configure TIBCO FTL on the same machine that you have installed

ActiveMatrix BusinessWorks 6.x on. For more information, see the "Setting Up TIBCO FTL® for BWAgent Transport" topic in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

> **Important:** For the version of TIBCO FTL that is supported with the version of ActiveMatrix BusinessWorks 6.x, see the ActiveMatrix BusinessWorks readme.

- Install Microsoft SQL server. The Microsoft SQL driver is available by default.

After installing Microsoft SQL Server, create a database bwadmindb and grant privileges to the default database owner root as described in the following methods:

SQL Server Authentication

Procedure

1. Open Microsoft SQL Server Management Studio.
2. From the **Object Explorer** pane, right-click **Databases > New Database** to create the database bwadmindb.
3. Right-click **Security > Logins > New Login...**.
4. Make the following configurations from the Login-New window:
    a. Type bwuser in the **Login name** field.
    b. Select **SQL Server Authentication**.
    c. **Optional.** Unselect **Enforce password policy**, **Enforce password expiration**, or **User must change password at next login**.
    d. In the **Default database** field, select **bwadmindb**.
    e. From the Select a page pane, on the left side of the Login - New window, click the **Server Roles** tab.
    f. To configure the BWAgent with MS SQL Server, set the values for Minimum Server Role and Database Role required for a user, for a particular database. In MS SQL Server Management Server, navigate to **Security > Logins**. Right-click **Login Properties > Server Roles**. The minimum server role required for a particular user is *public*. Under User Mapping, the minimum database role membership for the selected database for a user mapped to the login should

be one of the following two combinations: public and db_owner OR public, db_datawriter, db_datareader, and db_ddladmin.

g. Click **OK**.

5. Stop the BWAgent if it is running.
6. Open the `bwagent_ftl.json` file in `BW_HOME\config` (Windows) or `${BW_HOME}/config` (Unix).
7. Update the following properties for your environment:

Property Name	Microsoft SQL Value
`dbtype`	`sqlserver`
`dbdriver`	`com.microsoft.sqlserver.jdbc.SQLServerDriver`
`dbconnectionurl`	`jdbc:sqlserver://localhost:1433;databaseName=bwadmindb;trustServerCertificate=true`
`dbuser`	`bwuser`
`dbpassword`	`bwuser`

8. Run the BWAdmin `config` command with the `-cf` option and push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_ftl.json agent
```

9. Restart the BWAgent.

Windows Authentication

Procedure

1. Open Microsoft SQL Server Management Studio.
2. From the **Object Explorer** pane, right-click **Databases > New Database** to create the

database `bwadmindb`.

Ensure that you have the required permissions on the SQL server.

3. Configure the Database Engine with your system's Windows Authentication.
4. When using Windows Authentication Integrated Security, perform the following steps:
   a. Download the `sqljdbc_12.8.1.0_enu.zip` file from the official Microsoft website and extract it.
   b. Copy `sqljdbc_12.8.1.0_enu/enu/auth/x64/mssql-jdbc_auth-12.8.1.x64.dll` to `Java/jre7/bin` and `Java/jre7/lib`.
   c. Add the latest JAR file `mssql-jdbc-12.8.1.jre11.jar` to the following location: `/config/design/thirdparty`.
   d. Add the same latest JAR file to BWAgent Home: `\system\shared\com.tibco.bw.tpcl.jdbc.datasourcefactory.sqlserver_4.0.1300.012\lib`.
   e. Then, add the same latest JAR file in the Bundle-ClassPath of the `MANIFEST.MF` file.
5. Stop the BWAgent if it is running.
6. Open the `bwagent_ftl.json` file in `BW_HOME\config` (Windows) or `${BW_HOME}/config` (Unix).
7. Update the following properties for your environment.

Property Name	Microsoft SQL Value
`dbtype`	`sqlserver`
`dbdriver`	`com.microsoft.sqlserver.jdbc.SQLServerDriver`
`dbconnectionurl`	`jdbc:sqlserver://localhost:1433;databaseName=Testdb;IntegratedSecurity=true;encrypt=false`
`dbuser`	`systemUserName`
`dbpassword`	Blank

8. Run the BWAdmin `config` command with the `-cf` option and push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_ftl.json agent
```

9. Restart the BWAgent.

Configuring BWAgent for Oracle and TIBCO FTL®

The BWAgent can be configured to use the Oracle Database with TIBCO FTL for transport.

Note: Use of TIBCO FTL with ActiveMatrix BusinessWorks for configuring BWAgent and configuring group provider for BWEngine do not require TIBCO FTL licenses.

Note: The database name must be unique per agent network if multiple networks share the same physical database. BWAgent and BWEngine support the sharing of the same database.

Before you begin

Install and configure TIBCO FTL on the same machine where you have installed ActiveMatrix BusinessWorks 6.x. For more information, see the "Setting Up TIBCO FTL® for BWAgent Transport" topic in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

Important: For the version of TIBCO FTL that is supported with the version of ActiveMatrix BusinessWorks 6.x, see the ActiveMatrix BusinessWorks readme.

Ensure that you have installed FTL client libraries. For more information, see the "Integrating with TIBCO FTL" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

Install the latest Oracle Database:

1. Download and install the latest Oracle Database from their official website.
2. Configure the server configuration by following the prompts in the Oracle Configuration wizard.

3. Accept the default port value 1521, or enter your own port number.
4. Download the latest JDBC driver connector JAR files to the `BW_HOME\config\drivers\shells\jdbc.oracle.runtime\runtime\plugins\com.tibco.bw.jdbc.datasourcefactory.oracle\lib` folder for Windows or the `${BW_HOME}/system/lib` for Unix.
5. Install the Oracle driver by running the command `bwinstall oracle-driver` from the `/bin` folder.

Note: If you are using Oracle Database 11g, run the oracle11g_create.sql script at *BW_HOME*`/config/dbscripts/admin/oracle` and restart the BWAgent.

Procedure

1. After installing the Oracle Database, create a database `bwadmindb` and grant privileges to the default database owner `bwuser` as described in the following steps:
   a. Run the following commands in SQLPlus as a root user:

```
CREATE DATABASE bwadmindb;
create USER C##bwuser identified by "bwuser";
GRANT CREATE SESSION TO C##bwuser;
grant create sequence to C##bwuser;
ALTER USER C##bwuser quota unlimited on USERS;
grant create table to C##bwuser;
```

2. Stop the BWAgent if it is running.
3. Open the `bwagent_ftl.json` file in `BW_HOME\config` (Windows) or `${BW_HOME}/config` (Unix).
4. Update the following properties for your environment:

Property Name	Oracle Value
`dbtype`	`oracle`
`dbdriver`	`oracle.jdbc.OracleDriver`
`dbconnectionurl`	`jdbc:oracle:thin:@db:1521:bwadmindb`

Property Name	Oracle Value
dbuser	Cbwuser
dbpassword	bwuser

5. Run the BWAdmin `config` command with the `-cf` option to push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_ftl.json agent
```

> **Note:** If you are creating a user with '##' (for example, c##bwuser), then you need to keep the `username` field as empty in the `bwagent_db.json` file and later update the `bwagent.ini` file manually.

6. Restart the BWAgent.

Configuring BWAgent for DB2 and TIBCO FTL®

The BWAgent can be configured to use DB2 database with TIBCO FTL for transport.

> **Note:** Use of TIBCO FTL with ActiveMatrix BusinessWorks for configuring BWAgent and for configuring group provider for engine does not require TIBCO FTL licenses.

> **Note:** The database name must be unique per agent network if multiple networks share the same physical database.

Before you begin

- Install and configure TIBCO FTL on the same machine that you have installed ActiveMatrix BusinessWorks 6.x on. For more information, see the "Setting Up TIBCO FTL® for BWAgent Transport" topic in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

> **Important:** For the version of TIBCO FTL that is supported with the version of ActiveMatrix BusinessWorks 6.x, see the ActiveMatrix BusinessWorks readme.

- Download the latest DB2 server package from their official website and install DB2. Configure the server configuration by following the prompts in the DB2 Server Configuration wizard.
- Download the latest JDBC driver and connector JAR files for DB2 to the `BW_HOME\config\drivers\shells\jdbc.db2.runtime\runtime\plugins\com.tibco.bw.jdbc.datasourcefactory.db2\lib` folder for Windows or the `${BW_HOME}/system/lib` folder for Unix.
- Install the DB2 driver by running the command `bwinstall db2-driver` from the `/bin` folder.

Procedure

1. Log in to DB2 and create database by running the following command:

   ```
   CREATE DATABASE <database name> USING CODESET UTF-8 TERRITORY US
   COLLATE USING SYSTEM PAGESIZE 16384
   ```

   > **Note:** Set the page size to 16K (16384) or higher.

2. Stop the BWAgent if it is running.
3. Open the `bwagent_ftl.json` file in `BW_HOME\config` (Windows) or `${BW_HOME}/config` (Unix).
4. Update the following properties for your environment:

Property Name	DB2 Value
`dbtype`	db2
`dbdriver`	com.ibm.db2.jcc.DB2Driver
`dbconnectionurl`	jdbc:db2://localhost:50000/bwadmindb
`dbuser`	bwuser

Property Name	DB2 Value
dbpassword	bwuser

5. Run the BWAdmin `config` command with the `-cf` option to push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_ftl.json agent
```

6. Restart the BWAgent.

Configuring BWAgent for MariaDB and TIBCO FTL®

The BWAgent can be configured to use MariaDB database with TIBCO FTL for persistence and transport.

Note: The database name must be unique per agent network if multiple networks share the same physical database. BWAgent and BWEngine support sharing the same database, users, and schemas.

Before you begin

- Install the latest TIBCO Enterprise Messaging Server and start the server.
- Download the latest MariaDB server package from https://downloads.mariadb.org/ and install MariaDB. Configure the server configuration by following the prompts in the MariaDB Server Configuration wizard. Ensure that you select the following values:
  - Database: Multifunctional
  - Type of connectivity: Manual
  - Default port: 3306
- Download the latest JAR files for MariaDB to the `BW_HOME\config\drivers\shells\jdbc.mariadb.runtime\runtime\plugins\com.tibco.bw.jdbc.datasourcefactory.mariadb\lib` folder.
- Install the MariaDB driver by running the command `bwinstall mariadb-driver` from the `/bin` folder.

- Ensure you have installed EMS client libraries. For more information, see "Integrating with TIBCO Enterprise Message Service™" in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.
- Optional. If you are upgrading to ActiveMatrix BusinessWorks 6.3.2, or later versions of the software, upgrade the database schema. For more information, see "Updating the Database Schema to Enable BWAgent to Use Database with TIBCO Enterprise Message Service" in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

Procedure

1. Create a database bwadmindb and grant privileges to the default database owner root as described in the following steps:
   a. Run the following command on the MariaDB terminal:

      ```
      mariadb>create database bwadmindb
      ```

   b. Run the following command to view the tables included in the newly created database:

      ```
      mariadb>use bwadmindb;
      ```

   c. Run the following command to grant all privileges to the root user for that database after replacing the value for the host IP address:

      ```
      mariadb>GRANT ALL PRIVILEGES ON *.* TO root@<host_IP> IDENTIFIED
      BY 'Tibco123'WITH GRANT OPTION;
      ```

      To grant the minimum number of permissions to the bwuser for that database, run the following command, where *<host_IP>* is replaced with the value for the host IP address:

      ```
      grant create,select,update,insert,delete ON *.* to bwuser@<host_
      IP> IDENTIFIED BY "bwuser";
      ```

2. Stop the BWAgent if it is running.
3. Open the `bwagent_db.json` file in *BW_HOME*`\config` (Windows) or `${`*BW_HOME*`}/config` (Unix).
4. Update the following properties for your environment:

Property Name	MariaDB Value
`dbtype`	mariadb
`dbdriver`	org.mariadb.jdbc.Driver
`dbconnectionurl`	jdbc:mariadb://localhost:3306/bwadmindb
`dbuser`	bwuser
`dbpassword`	bwuser

5. Run the BWAdmin `config` command with the `-cf` option to create an `.ini` file in the correct location.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

6. Restart the BWAgent.

Database with TIBCO Enterprise Message Service™ Configuration for BWAgent

The BWAgent can be configured to use an external relational database for persistence and the TIBCO Enterprise Message Service™ (EMS) for transport among BWAgents. PostgreSQL, MySQL, Microsoft SQL, and Oracle are the supported databases.

In your production environment, ensure you are using an external database for data persistence and either TIBCO FTL® or TIBCO Enterprise Message Service™ for communication transport.

Note: Regularly back up domain data using the BWAdmin `backup` command. For more information about backing up and restoring domain data, see Backing up and Restoring a Domain.

Note: For a non-admin EMS user, the following permissions are to be given to use BWAgent:

```
grant topic "$sys.monitor.>" user=<username> all
grant admin user=<username> view-connection,view-server,create-
destination
```

Note: Make sure that the TIBCO Enterprise Message Service™ (EMS) server has a dynamic permission for topics or queues. For more information, see the "Wildcards and Dynamically Created Destinations" topic in the *TIBCO Enterprise Message Service™* guide.

For a multi-agent, multi-machine environment using an external database and TIBCO Enterprise Message Service, the following properties in the `bwagent.ini` file are important.

BWAgent properties for Multi-Agent, Multi-Machine Environments using Database/EMS

Property Name	Description
`bw.agent.technology.dbems.db.provider`	The database provider such as: • `postgresql` • `mysql` • `mssql` • `oracle` • `db2`
`bw.agent.technology.dbems.db.driver`	The database driver. Example: `dbDriver=org.postgresql.Driver`
`bw.agent.technology.dbems.db.connectionURL`	The URL to connect to the database. Example: `jdbc:postgresql://localhost:5432/bwadmindb`
`bw.agent.technology.dbems.db.userName`	The username to authenticate to the database.

Property Name	Description
bw.agent.technology.dbems.db.password	The password to authenticate to the database.
bw.agent.technology.dbems.ems.serverUrl	The URL to connect to the EMS server. Example: tcp://localhost:7222
bw.agent.technology.dbems.ems.userName	The username to authenticate to the EMS server. The default is admin. To authenticate a non-admin user, create a user and set the password. Run the following two commands in the TIBCO EMS admin console: • grant topic "$sys.monitor.>" user=sri2 all • grant admin user=sri2 view-connection,view-server
bw.agent.technology.dbems.ems.password	The password to authenticate to the EMS server. There is no password by default. You can provide the obfuscated password. For more information about how to obfuscate passwords, see Obfuscating or Encrypting Password for Database, EMS, and FTL Users.
bw.agent.technology.dbems.ems.requestQueueName	Member Queue Name. Set the value as bw6.admin.operations.queue.<memberQueueName> where memberQueueName is the value of bw.agent.memberName. For example, bw.agent.memberName=machine1, bw.agent.technology.dbems.ems.requestQueueName=bw6.admin.operations.queue.machine1 **Note:** When creating an agent network, create and use the separate EMS queue name for each member.
bw.agent.technology.requestTimeout	Timeout for requests sent to other BWAgents The default value is 60000 milliseconds.

Property Name	Description
`bw.agent.technology.remote.status.requestTimeout`	Timeout for requests sent to BWAgents to find the status of AppNodes, applications, and other BWAgents. The default value is 3000 milliseconds.
`bw.agent.technology.dbems.ems.reconnection.interval`	Set the `bw.agent.technology.dbems.ems.reconnection.interval` property to specify, in milliseconds, how often the BWAgent checks its connection with the EMS server. The default value is 10000 milliseconds.

For information about setting properties, see:

- PostgreSQL - For instructions, see Configuring BWAgent for PostgreSQL and TIBCO Enterprise Message Service.
- MySQL - For instructions, see Configuring BWAgent for MySQL and TIBCO Enterprise Message Service.
- Microsoft SQL - For instructions, see Configuring BWAgent for Microsoft SQL Server and TIBCO Enterprise Message Service.
- Oracle - For instructions, see Configuring BWAgent for Oracle and TIBCO Enterprise Message Service.
- DB2 - For instructions, see Configuring BWAgent for DB2 and TIBCO Enterprise Message Service.
- MariaDB - For instructions, see Configuring BWAgent for MariaDB and TIBCO Enterprise Message Service.

Configuring BWAgent for PostgreSQL and TIBCO Enterprise Message Service

The BWAgent can be configured to use the PostgreSQL database with the TIBCO Enterprise Message Service (EMS) for persistence and transport.

Note: The database name must be unique per agent network if multiple networks share the same physical database. BWAgent and BWEngine support sharing the same database, users, and schemas.

Before you begin

- Install the latest TIBCO Enterprise Messaging Server and start the server.
- Install PostgreSQL. The PostgreSQL driver is available by default.
- Install EMS client libraries. For more information, see the "Integrating with TIBCO Enterprise Message Service™" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.
- (Optional) If you are upgrading to ActiveMatrix BusinessWorks 6.3.2, or later versions of the software, upgrade the database schema. For more information, see the "Updating the Database Schema to Enable BWAgent to Use Database with TIBCO Enterprise Message Service" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

Procedure

1. After installing PostgreSQL, create a `bwadmindb` database and the database owner, `bwuser`, as described in the following steps:
    a. Run the following commands on the psql terminal:

    ```
    > psql -p 5432 -c "CREATE USER bwuser WITH CREATEDB PASSWORD
    'bwuser';"
    > psql -p 5432 -c "CREATE DATABASE bwadmindb WITH OWNER
    bwuser;"
    ```

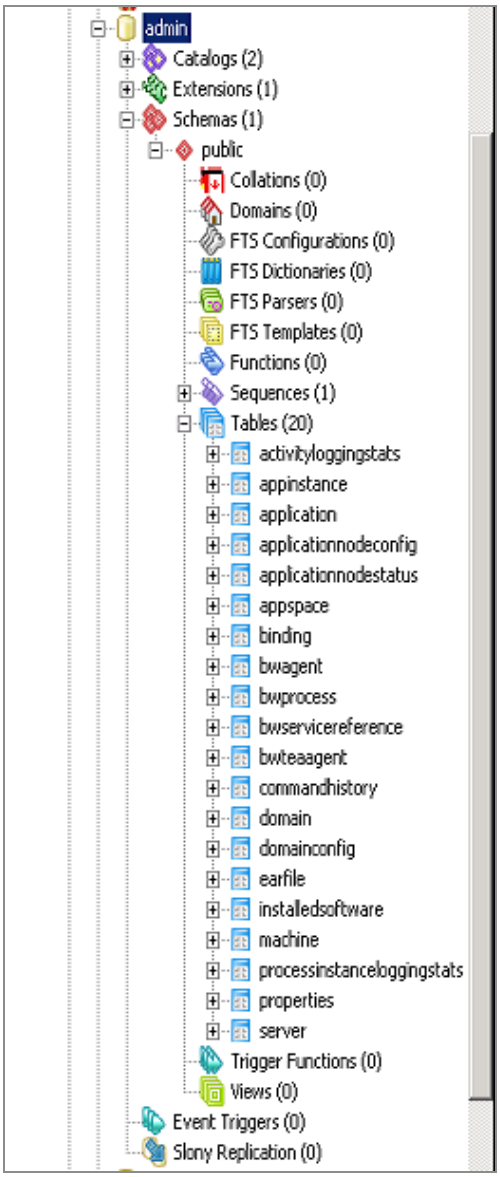
    b. To add a password for the database owner, expand **Login Roles > bwuser**, right-click, and select **Properties**. Click the **Definition** tab and create and save a password.
2. Stop the BWAgent if it is running.
3. Open the `bwagent_db.json` file in *BW_HOME*`\config` (Windows) or `${`*BW_HOME*`}/config` (Unix).
4. Update the following properties for your environment:

Property Name	PostgreSQL Value
dbtype	postgresql
dbdriver	org.postgresql.Driver
dbconnectionURL	jdbc:postgresql://localhost:5432/bwadmindb
dbuser	bwuser
dbpassword	bwuser

5. Run the BWAdmin `config` command with the `-cf` option and push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

6. Restart the BWAgent.
7. Open the `pgAdmin III` utility and expand **Schemas > Tables** in the **Object Browser** to view the tables in the database:



Configuring BWAgent for MySQL and TIBCO Enterprise Message Service

The BWAgent can be configured to use MySQL database with TIBCO Enterprise Message Service (EMS) for persistence and transport.

Note: The database name must be unique per agent network if multiple networks share the same physical database. BWAgent and BWEngine supports sharing the same database, users, and schemas.

Before you begin

- Install the latest TIBCO Enterprise Messaging Server and start the server.
- Download the latest MySQL server package from their official website and install MySQL. Configure the server configuration by following the prompts in the MySQL Server Configuration wizard. Ensure that you select the following values:
    - Database: Multifunctional
    - Type of connectivity: Manual
    - Default port: 3306
- Download the latest connector JAR files for MySQL to the `BW_HOME\config\drivers\shells\jdbc.mysql.runtime\runtime\plugins\com.tibco.bw.jdbc.datasourcefactory.mysql\lib` folder.
- Install the MySQL driver by running the `bwinstall mysql-driver` command from the `/bin` folder.
- Ensure you have installed EMS client libraries. For more information, see the "Integrating with TIBCO Enterprise Message Service™" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide for additional details.
- (Optional) If you are upgrading to ActiveMatrix BusinessWorks 6.3.2, or later versions of the software, upgrade the database schema. For more information, see the "Updating the Database Schema to Enable BWAgent to Use Database with TIBCO Enterprise Message Service" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

Procedure

1. Create a `bwadmindb` database and grant privileges to the default database owner `root` as described in the following steps:
    a. Run the following command on the MySQL terminal:

    For MySQL 5.x

    ```
    mysql>create database bwadmindb;
    ```

    For MySQL 8.x

```
CREATE SCHEMA `bwadmindb` DEFAULT CHARACTER SET utf8;
```

b. Run the following command to view the tables included in the newly created database:

```
mysql>use bwadmindb;
```

c. Run the following command to grant all privileges to the root user for that database after replacing the value for the host IP address:

```
mysql>GRANT ALL PRIVILEGES ON *.* TO 'root'@<host_IP> IDENTIFIED
BY 'Tibco123'WITH GRANT OPTION;
```

To grant the minimum number of permissions to the bwuser for that database, run the following command, where *<host_IP>* is replaced with the value for the host IP address:

```
grant create,select,update,insert,delete ON *.* to
'bwuser'@<host_IP> IDENTIFIED BY 'bwuser';
```

2. Stop the BWAgent if it is running.
3. Open the `bwagent_db.json` file in *BW_HOME*`\config` (Windows) or `${`*BW_HOME*`}/config` (Unix).
4. Update the following properties for your environment:

Property Name	MySQL Value
`dbtype`	mysql
`dbdriver`	com.mysql.jdbc.Driver
`dbconnectionurl`	jdbc:mysql://localhost:3306/bwadmindb?useSSL=false
`dbuser`	bwuser
`dbpassword`	bwuser

5. Run the BWAdmin `config` command with the `-cf` option to create an `.ini` file in the correct location.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

6. Restart the BWAgent.

Configuring BWAgent for Microsoft SQL Server and TIBCO Enterprise Message Service

The BWAgent can be configured to use Microsoft SQL Server database with TIBCO Enterprise Message Service (EMS) for persistence and transport.

Note: The database name must be unique per agent network if multiple networks share the same physical database. BWAgent and BWEngine support sharing the same database, users, and schemas.

Before you begin

- Install the latest TIBCO Enterprise Messaging Server and start the server.
- The Microsoft SQL driver is available by default.
- Ensure you have installed EMS client libraries. For more information, see the "Integrating with TIBCO Enterprise Message Service™" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.
- (Optional) If you are upgrading to ActiveMatrix BusinessWorks 6.3.2, or later versions of the software, upgrade the database schema. For more information, see the "Updating the Database Schema to Enable BWAgent to Use Database with TIBCO Enterprise Message Service" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide for additional details.

After installing Microsoft SQL Server, create a database `bwadmindb` and grant privileges to the default database owner `root` as described in the following methods:

SQL Server Authentication

Procedure

1. Open Microsoft SQL Server Management Studio.
2. From the **Object Explorer** pane, right-click **Databases > New Database** to create the database `bwadmindb`.
3. Right-click **Security > Logins > New Login...**.
4. Make the following configurations from the Login-New window:
    a. Type `bwuser` in the **Login name** field.
    b. Select **SQL Server Authentication**.
    c. (Optional) Clear the **Enforce password policy**, **Enforce password expiration**, or **User must change password at next login** checkboxes.
    d. In the **Default database** field, select **bwadmindb**.
    e. From the Select a page pane, on the left side of the Login - New window, click the **Server Roles** tab.
    f. To configure the BWAgent with MS SQL Server, set the values for Minimum Server Role and Database Role required for a user, for a particular database. In MS SQL Server Management Server, navigate to **Security > Logins**. Right-click **Login Properties** > **Server Roles**. The minimum server role required for a particular user is *public*. Under User Mapping, the minimum database role membership for the selected database for a user mapped to the login should be one of the following two combinations: public and db_owner OR public, db_datawriter, db_datareader, and db_ddladmin.
    g. Click **OK**.
5. Stop the BWAgent if it is running.
6. Open the `bwagent_db.json` file in `BW_HOME\config` (Windows) or `${BW_HOME}/config` (Unix).
7. Update the following properties for your environment:

Property Name	Microsoft SQL Value
`dbtype`	`sqlserver`

Property Name	Microsoft SQL Value
dbdriver	com.microsoft.sqlserver.jdbc.SQLServerDriver
dbconnection url	jdbc:sqlserver://localhost:1433;databaseName=bwadmindb;trustServerCertificate=true
dbuser	bwuser
dbpassword	bwuser

8. Run the BWAdmin `config` command with the `-cf` option and push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

9. Restart the BWAgent.

Windows Authentication

Procedure

1. Open Microsoft SQL Server Management Studio.
2. From the **Object Explorer** pane, right-click **Databases > New Database** to create the database `bwadmindb`.

   Ensure that you have the required permissions on the SQL server.
3. Configure the Database Engine with your system's Windows Authentication.
4. When using Windows Authentication Integrated Security, perform the following steps:
   a. Download the `sqljdbc_12.8.1.0_enu.zip` file from the official Microsoft website and extract it.
   b. Copy `sqljdbc_12.8.1.0_enu/enu/auth/x64/mssql-jdbc_auth-12.8.1.x64.dll` to `Java/jre7/bin` and `Java/jre7/lib`.
   c. Add the latest JAR file `mssql-jdbc-12.8.1.jre11.jar` to the following location:

`/config/design/thirdparty`.

d. Add the same latest JAR file to BWAgent Home: `\system\shared\com.tibco.bw.tpcl.jdbc.datasourcefactory.sqlserver_4.0.1300.012\lib`.

e. Then add the same latest JAR file in the Bundle-ClassPath of the `MANIFEST.MF` file.

5. Stop the BWAgent if it is running.
6. Open the `bwagent_db.json` file in `BW_HOME\config` (Windows) or `${BW_HOME}/config` (Unix).
7. Update the following properties for your environment.

Property Name	Microsoft SQL Value
`dbtype`	`sqlserver`
`dbdriver`	`com.microsoft.sqlserver.jdbc.SQLServerDriver`
`dbconnectionurl`	`jdbc:sqlserver://localhost:1433;databaseName=Testdb;IntegratedSecurity=true;encrypt=false`
`dbuser`	`systemUserName`
`dbpassword`	Blank

8. Run the BWAdmin `config` command with the `-cf` option and push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

9. Restart the BWAgent.

Configuring BWAgent for Oracle and TIBCO Enterprise Message Service

The BWAgent can be configured to use the Oracle Database with TIBCO Enterprise Message Service (EMS) for persistence and transport.

Note: The database name must be unique per agent network if multiple networks share the same physical database. BWAgent and BWEngine support sharing the same database.

Before you begin

- Install the latest TIBCO Enterprise Messaging Server and start the server.
- Download and install the latest Oracle Database from their official website.
- Configure the server configuration by following the prompts in the Oracle Configuration wizard.
- Accept the default port value (1521), or enter your own port number.
- Download the latest JDBC driver connector JAR files to the `BW_HOME\config\drivers\shells\jdbc.oracle.runtime\runtime\plugins\com.tibco.bw.jdbc.datasourcefactory.oracle\lib` folder.
- Install the Oracle driver by running the command `bwinstall oracle-driver` from the `/bin` folder.

  Note: If you are using Oracle Database 11g, run the oracle11g_create.sql script at `BW_HOME/config/dbscripts/admin/oracle` and restart the BWAgent.

- Ensure you have installed EMS client libraries. For more information, see the "Integrating with TIBCO Enterprise Message Service™" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.
- Optional. If you are upgrading to ActiveMatrix BusinessWorks 6.3.2, or later versions of the software, upgrade the database schema. For more information, see the "Updating the Database Schema to Enable BWAgent to Use Database with TIBCO Enterprise Message Service" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

Procedure

1. After installing the latest Oracle Database, create a `bwadmindb` database and grant privileges to the default database owner, `bwuser` as described in the following steps:
   a. Run the following commands in SQLPlus as a root user:

```
CREATE DATABASE bwadmindb;
create USER C##bwuser identified by "bwuser";
GRANT CREATE SESSION TO C##bwuser;
grant create sequence to C##bwuser;
ALTER USER C##bwuser quota unlimited on USERS;
grant create table to C##bwuser;
```

2. Stop the BWAgent if it is running.
3. Open the `bwagent_db.json` file in `BW_HOME\config` (Windows) or `${BW_HOME}/config` (Unix).
4. Update the following properties for your environment:

Property Name	Oracle Value
`dbtype`	`oracle`
`dbdriver`	`oracle.jdbc.OracleDriver`
`dbconnectionurl`	`jdbc:oracle:thin:@localhost:1521:bwadmindb`
`dbuser`	`Cbwuser`
`dbpassword`	`bwuser`

5. Run the BWAdmin `config` command with the `-cf` option push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

> **Note:** If you are creating a user with '##' (for example, c##bwuser), then you need to keep the `username` field as empty in the `bwagent_db.json` file and later update the `bwagent.ini` file manually.

6. Restart the BWAgent.

Configuring BWAgent for DB2 and TIBCO Enterprise Message Service

The BWAgent can be configured to use the DB2 database with TIBCO Enterprise Message Service (EMS) for persistence and transport.

Note: If multiple networks share the same physical database, the database name must be unique per agent network.

Before you begin

- Install the latest TIBCO Enterprise Messaging Server and start the server.
- Download the latest DB2 server package from their official website and install DB2. Configure the server configuration by following the prompts in the DB2 Server Configuration wizard.
- Download the latest JDBC driver and connector JAR files for DB2 to the `BW_HOME\config\drivers\shells\jdbc.db2.runtime\runtime\plugins\com.tibco.bw.jdbc.datasourcefactory.db2\lib` folder.
- Install the DB2 driver by running the command `bwinstall db2-driver` from the `/bin` folder.
- Ensure you have installed EMS client libraries. For more information, see the "Integrating with TIBCO Enterprise Message Service™" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.
- Optional. If you are upgrading to ActiveMatrix BusinessWorks 6.3.2, or later versions of the software, upgrade the database schema. For more information, see the "Updating the Database Schema to Enable BWAgent to Use Database with TIBCO Enterprise Message Service" section in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

Procedure

1. Log in to DB2 and create a database by running the following command:

```
CREATE DATABASE <database name> USING CODESET UTF-8 TERRITORY US
COLLATE USING SYSTEM PAGESIZE 16384
```

> **Note:** Set the page size to 16K (16384) or higher.

2. Stop the BWAgent if it is running.
3. Open the `bwagent_db.json` file in `BW_HOME\config` (Windows) or `${BW_HOME}/config` (Unix).
4. Update the following properties for your environment:

Property Name	DB2 Value
`dbtype`	`db2`
`dbdriver`	`com.ibm.db2.jcc.DB2Driver`
`dbconnectionurl`	`jdbc:db2://localhost:50000/bwadmindb`
`dbuser`	`bwuser`
`dbpassword`	`bwuser`

5. Run the BWAdmin `config` command with the `-cf` option and push the changes from the JSON file to the `bwagent.ini` file.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

6. Restart the BWAgent.

Configuring BWAgent for MariaDB and TIBCO Enterprise Message Service

The BWAgent can be configured to use the MariaDB database with TIBCO Enterprise Message Service (EMS) for persistence and transport.

> **Note:** The database name must be unique per agent network if multiple networks share the same physical database. BWAgent and BWEngine support sharing the same database, users, and schemas.

Before you begin

- Install the latest TIBCO Enterprise Messaging Server and start the server.
- Download the latest MariaDB server package from https://downloads.mariadb.org/ and install MariaDB. Configure the server configuration by following the prompts in the MariaDB Server Configuration wizard. Ensure that you select the following values:
  - Database: Multifunctional
  - Type of connectivity: Manual
  - Default port: 3306
- Download the latest JAR files for MariaDB to the `BW_HOME\config\drivers\shells\jdbc.mariadb.runtime\runtime\plugins\com.tibco.bw.jdbc.datasourcefactory.mariadb\lib` folder.
- To install the MariaDB driver, run the command `bwinstall mariadb-driver` from the `/bin` folder.
- Ensure you have installed EMS client libraries. For more information, see Integrating with TIBCO Enterprise Message Service™ in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.
- Optional. If you are upgrading to ActiveMatrix BusinessWorks 6.3.2, or later versions of the software, upgrade the database schema. For more information, see "Updating the Database Schema to Enable BWAgent to Use Database with TIBCO Enterprise Message Service" in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

Procedure

1. Create a `bwadmindb` database and grant privileges to the default database owner `root` as described in the following steps:
   a. Run the following command on the MariaDB terminal:

      ```
      mariadb>create database bwadmindb
      ```

   b. Run the following command to view the tables included in the newly created database:

      ```
      mariadb>use bwadmindb;
      ```

c. Run the following command to grant all privileges to the root user for that database after replacing the value for the host IP address:

```
mariadb>GRANT ALL PRIVILEGES ON *.* TO root@<host_IP> IDENTIFIED
BY 'Tibco123'WITH GRANT OPTION;
```

To grant the minimum number of permissions to the bwuser for that database, run the following command, where *<host_IP>* is replaced with the value for the host IP address:

```
grant create,select,update,insert,delete ON *.* to bwuser@<host_
IP> IDENTIFIED BY "bwuser";
```

2. Stop the BWAgent if it is running.
3. Open the `bwagent_db.json` file in `BW_HOME\config` (Windows) or `${BW_HOME}/config` (Unix).
4. Update the following properties for your environment:

Property Name	MariaDB Value
`dbtype`	`mariadb`
`dbdriver`	`org.mariadb.jdbc.Driver`
`dbconnectionurl`	`jdbc:mariadb://localhost:3306/bwadmind`
`dbuser`	`bwuser`
`dbpassword`	`bwuser`

5. Run the BWAdmin `config` command with the `-cf` option to create an `.ini` file in the correct location.

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

6. Restart the BWAgent.

Obfuscating or Encrypting Password for Database, EMS, and FTL Users

By default, the Database, EMS, and FTL users do not have a password. You can set the password; however, this password is not encrypted.
To obfuscate the password, perform the following steps using Admin CLI.

Procedure

1. Generate an obfuscated password using BWAdmin utility.

   ```
   bwadmin[admin]> obfuscate <user password>
   ```

   It shows the generated obfuscated password.
2. Use this obfuscated password in the `bwagent.ini` file.

Creating an Agent Network

This topic shows how to configure BWAgents so that they can be members of the same agent network.

When using multiple machines, the runtime status of the BWAgents and AppNodes cannot be computed reliably if the machine clocks in the agent network are not in sync with each other. Make sure that the clocks for machines in the network are synchronized.

Note: All agents in a network should be of the same 4-part version.

For example, If there is one agent with version 6.4.2_HF009, all the other agents should be of the version 6.4.2_HF009 only.

Complete the following steps for each BWAgent that is to join the agent network.

Procedure

1. Stop BWAgent.
2. For each BWAgent, open the JSON configuration file, in *BW_HOME*`\config` (Windows) or `${`*BW_HOME*`}/config` (Unix). Use the configuration file specific to the technology type used by the BWAgents in the network.

a. Edit the parameters in this file as follows:

Parameter	Property in `bwagent.ini` File	Setting
`bwagentnetworkname`	`bw.agent.network.name`	The name of the network. Must be the same setting for each BWAgent in the network.
	`bw.agent.technology.dbems.ems.serverUrl/bw.agent.technology.dbftl.ftl.realmserver`	Use the same transport layer URL
	`bw.agent.technology.dbems.db.connectionURL` and `bw.agent.technology.dbftl.db.connectionURL`	Same database

b. Set other parameters in the JSON file. For more information about parameters, see Configuring BWAgent.

c. Save the file and use the BWAdmin `config` command to push the changes from the JSON file to the `bwagent.ini` file.

   Use the `bwagent_db.json` or `bwagent_ftl.json` file as follows:

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_ftl.json agent
```

3. Restart BWAgent.

Result

Use the `show agents` command to show all discovered BWAgents. Agents in a network can be managed by any other BWAgent.

Accessing the BWAgent REST API with the Swagger UI

Use the Swagger UI to access the BWAgent REST API, where you can try out operations and see results using sample data.

Procedure

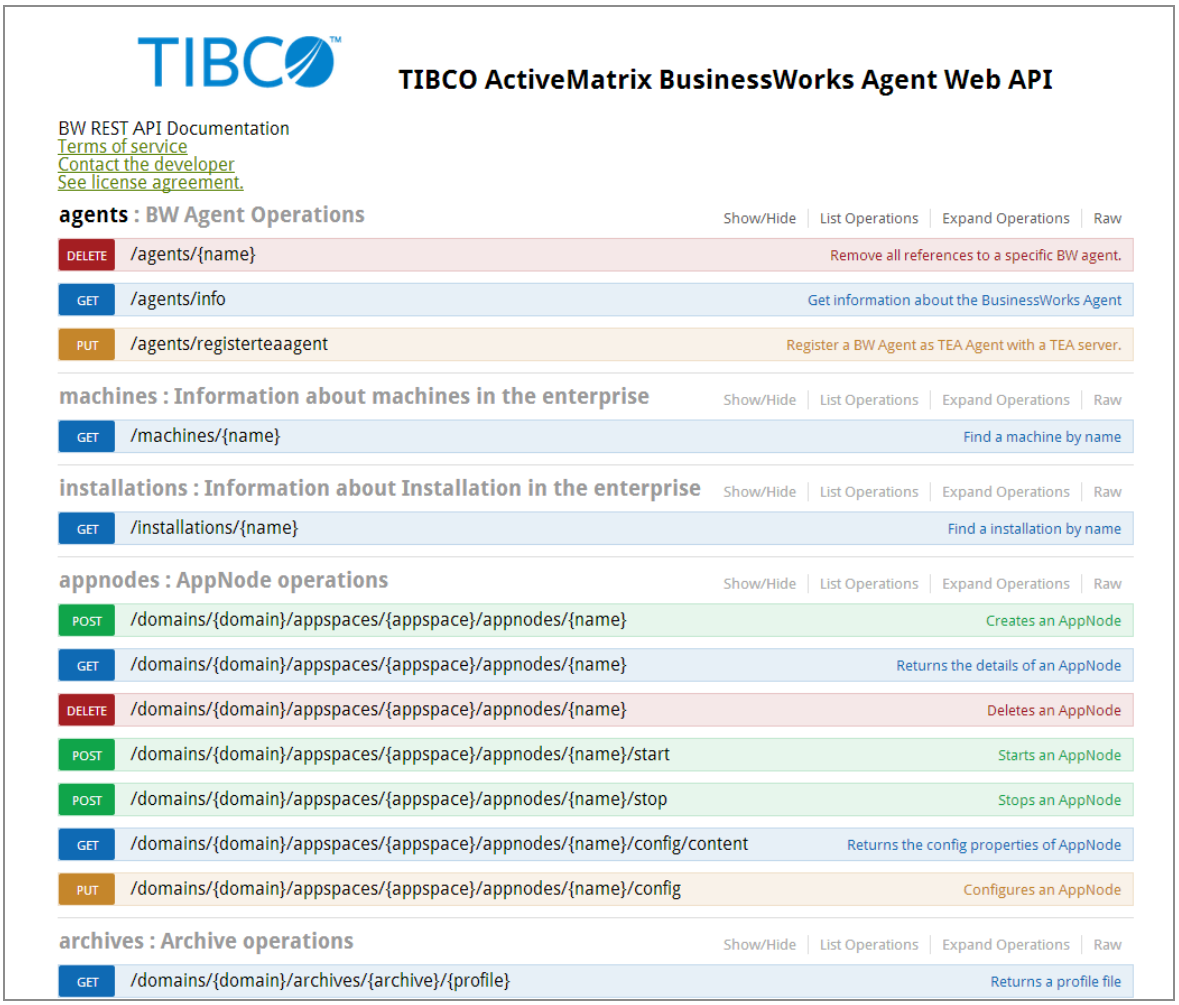
1. Start the BWAgent with the `apiserver` command:

```
BW_HOME\bin>bwagent apiserver
```

   The API server is started at `http://<hostname>:5555`, where <*hostname*> is the value you have set for the **bw.agent.http.host** property in the `bwagent.ini` file. For example, if you set `localhost` as the value for the **bw.agent.http.host** property, the API server is started at the following URL: `http://localhost:5555`.

2. Open a web browser and go to the URL of the API server.

   The BWAgent API documentation is displayed in the Swagger UI:



3. View sample data in the browser. To obtain the URL, go to the URL returned by an operation. For example, clicking **Try it out!** for the `GET/agents/info` operation returns the Request URL of `http://localhost:5555/api/agents/info`. Pasting this URL into the browser returns information similar to:

```
[{"name":"localhost","state":"Running","version":"6.6.0","configSta
te":"InSync","machineName":"bwin2k12r264b-76","description":"TIBCO
ActiveMatrix BusinessWorks version 6.6.0, build V37, 2019-10-
13","adminMode":"enterprise","tibcoHome":"E:\\BW6\\6.6.0\\V37","pid
":"5024","installationName":null,"configMap":null,"httpPort":null,"
httpHost":null,"uptime":10986,"internalPort":null}]
```

Note: To get actual data for the agent, go to the URL using port 8079 instead of 5555. Change the `api` folder in the file path to `bw` and the version number into the path before the resource. (The BWAgent must be running.) For example, the URL `http://localhost:8079/bw/v1/agents/info` returns the following information for an agent named MACHINE_1:


```
[{"name":"localhost","state":"Running","description":"TIBCO
ActiveMatrix BusinessWorks version 6.6.0, build V37, 2019-10-
13","tibcoHome":"E:\\BW6\\6.6.0\\V37","pid":"3356","configState":"I
nSync","machineName":"localhost","adminMode":"enterprise","version"
:"6.6.0","installationName":"V37","configMap":
{"bw.agent.technology.requestTimeout":"60000","bw.agent.technology.
dbftl.ftl.dataformat":"bw-
format","bw.agent.technology.type":"dbftl","bw.governance.jms.ssl.t
rust.store.location":"","bw.agent.tea.agent.port":"9091","bw.agent.
technology.dbftl.db.password":"bwpassword","bw.agent.technology.dbf
tl.db.driver":"com.mysql.jdbc.Driver","bw.monitor.ftluserpassword":
"","bw.governance.jms.ssl.trust.store.type":"JKS","bw.monitor.data.
format":"bytestream","bw.agent.technology.dbftl.db.connectionURL":"
jdbc:mysql://localhost:3306/V37","bw.monitor.ftlidentifier":"","bw.
agent.tea.server.url":"http://%HOSTADDRESS%:8777/tea","bw.monitor.f
tlendpoint":"bwadmin-stats-
endpoint","bw.agent.technology.statsProvider.db.driver":"com.mysql.
jdbc.Driver","bw.agent.http.port":"8079","bw.monitor.ftlinbox":"bw-
inbox","bw.governance.jms.server.password":"","bw.agent.technology.
statsProvider.db.connectionURL":"jdbc:mysql://localhost:3306/V37","
bw.monitor.ftldataformat":"bw-
format","bw.agent.technology.statsProvider":"db","bw.agent.http.hos
t":"0.0.0.0","bw.monitor.ftlapplicationname":"bwadminstats","bw.age
nt.technology.dbftl.ftl.inbox":"bw-
inbox","bw.agent.http.access.log.config":"bwagent-
access.xml","bw.admin.mode":"enterprise","bw.agent.technology.dbftl
.ftl.endpoint":"bwadmin-
endpoint","bw.governance.jms.server.url":"tcp://localhost:7222","bw
.agent.technology.dbftl.ftl.secondary":"","bw.agent.appnode.passwor
d":"OBF:1sho1wgi1u9d1x1d1xfj1x191ua51wfg1shu","bw.agent.technology.
dbftl.ftl.username":"","bw.governance.jms.reconnect.attempt.delay":
"500","bw.agent.memberName":"localhost","bw.governance.jms.server.u
sername":"admin","bw.monitor.ftlusername":"","bw.agent.technology.d
bftl.db.userName":"bwuser","bw.monitor.ftlsecondaryurl":"","bw.agen
```

```
t.technology.dbftl.db.provider":"mysql","bw.agent.technology.dbftl.
ftl.realmserver":"http://localhost:8070","bw.monitor.provider":"UDP
","bw.monitor.ftlrealmserverurl":"http://ip
[:port]","bw.governance.jms.reconnect.attempt.timeout":"500","bw.ag
ent.technology.statsProvider.db.userName":"bwuser","bw.governance.j
ms.queue.pd.receiver.name":"governance.de.bw.default","bw.agent.tea
.agent.host":"0.0.0.0","bw.agent.technology.dbftl.ftl.password":"",
"bw.agent.technology.db.create.schema":"true","bw.governance.jms.re
connect.attempt.count":"120","bw.governance.jms.ssl.trust.store.pas
sword":"","bw.agent.networkName":"BW6Network","bw.governance.enable
d":"false","bw.agent.technology.statsProvider.db.provider":"mysql",
"bw.agent.technology.statsProvider.db.password":"bwpassword","bw.ag
ent.technology.dbftl.ftl.application":"bwadmin","bw.agent.technolog
y.dbftl.ftl.identifier":""},"httpPort":8079,"httpHost":"0.0.0.0","u
ptime":11850866,"internalPort":56565}]
```


To change the URL interface or port, edit the `http.host` or `http.port` settings in the `bwagent.ini` file.

Using the BWAgent REST API to Return Selected Fields

You can retrieve information only of the selected fields by adding query parameters to the request URL.

The following sample queries show how to retrieve selected fields:

- **Example 1**: To check the status of an Application, the REST API GET URL would be -

  `http://localhost:8079/bw/v1/domains/<DomainName>/appspaces/<AppspaceName>/applications/<ApplicationName>/<ApplicationVersion>?fields=state`

  The URL returns the following response:

  {"state":"Running"}

  > **Note:** Use comma-separated fields after the question mark (?) with fields=keyword in the request query. Spaces are not permitted.

- **Example 2**: To get AppSpace details

  Normal query -

`http://localhost:8079/bw/v1/domains/<DomainName>/appspaces/<AppSpaceName>`. It returns the total payload, that is, all fields.

- **Select Query**: `http://localhost:8079/bw/v1/domains/<DomainName>/appspaces/<AppSpaceName>?select=name,status`. It retrieves only 2 fields.
- **Subfield**: `http://localhost:8079/bw/v1/domains/<DomainName>/appspaces/<AppSpaceName>?select=appSpaceConfigRefs.href`. It fetches href of appSpaceConfigRefs.

> **Note:** The query returns an empty response when the selected field is a collection.

Securing the BWAgent REST API

The BWAgent REST API can be secured via authentication and roles. The BWAgent REST API server can be secured with SSL access.

Enabling Authentication for the BWAgent REST API Using the JAAS Property File

Authentication for the BWAgent REST API is implemented using the JAAS property file login module. Different login module implementations can be used. For more information, see the Jetty documentation at eclipse.org.

Before you begin

Stop the BWAgent if it is running.

Procedure

1. To enable authentication for the BWAgent REST API, perform the following steps:
   a. Navigate to `BW_HOME\bin`.
   b. Open the `bwagent.tra` file in an editor and uncomment the following property:

```
java.property.java.security.auth.login.config=%BW_
```

```
HOME%/config/jaas.login.conf
```

This property points to the location of the JAAS configuration file, which enables the authentication mechanism and, in turn, points to the property file. The property file identifies users, and stores encrypted/obfuscated passwords.

c. Save the file.

2. To change the name or location of the default property file, edit the `BW_HOME\config\jaas.login.conf` file. The default property file for the login module implementation is `BW_HOME\config\realm.properties`.

```
bwloginmodule {
  org.eclipse.jetty.jaas.spi.PropertyFileLoginModule required
    file="../config/realm.properties";
};
```

3. To change default users or passwords, edit the `BW_HOME\config\realm.properties` file. Two users (**admin** and **bwappnode**) are provided by default. The **admin** user is the default user for accessing the BWAgent REST API. The **bwappnode** user is the default user for the BWAppNode process.

a. Open `BW_HOME\config\realm.properties` in a text editor. The default contents of the file are outlined below.

Note: The format of this file is `<username>: <password> [,<rolename> ...]`

```
admin: CRYPT:adpexzg3FUZAk,admin
bwappnode: OBF:1sho1wgi1u9d1x1d1xfj1x191ua51wfg1shu,admin
```

> **Note:** Roles specified in the **realm.properties** file are different than the TIBCO ActiveMatrix BusinessWorks™ user roles in TIBCO Enterprise Administrator™. For more information about TIBCO Enterprise Administrator user roles for ActiveMatrix BusinessWorks, see Roles and Permissions.

b. To create an encrypted or obfuscated password, navigate to the `BW_HOME\system\lib\tea` folder and use Java with the **-cp** option to call the Jetty Password utility, passing the password. In this example, the password **admin** is used. The utility returns the password in a variety of formats.

```
BW_HOME\system\lib\tea> java -cp jetty-util-<version>.jar
org.eclipse.jetty.util.security.Password admin password
2019-10-10 14:19:57.505:INFO::main: Logging initialized @281ms
to org.eclipse.jetty.util.log.StdErrLog
password
OBF:1v2j1uum1xtv1zej1zer1xtn1uvk1v1v
MD5:5f4dcc3b5aa765d61d8327deb882cf99
CRYPT:advwtv/9yU5yQ
```

c. Copy the OBF, MD5, or CRYPT password from the output and paste it into the `realm.properties` file for the username.

d. Save the file.

4. To configure the BWAppNode user using the BWAgent file,

   a. Open the `BW_HOME\config\bwagent.ini` file in a text editor and navigate to the **Configuration for AppNode to agent communication** section.

      This provides an alternate configuration mechanism.

   b. Uncomment the **bw.agent.appnode.user** property and change the default BWAppNode username. If you specify a different user for the **bw.agent.appnode.user** property, you can add the entry to the `realm.properties` file (with an obfuscated password and role). If you do not add this entry to the properties file, set the password for the user, outlined in the next step.

   c. Uncomment the **bw.agent.appnode.password** property and change the

default password. The password must be able to de-obfuscate. If encrypted, the BWAppNode process is unable to decrypt the password. As a result, the AppNode is unable to communicate with the BWAgent REST API and report its status. The status for the AppNode and any applications on that AppNode is displayed as Stopped even though they are running. To create an encrypted or obfuscated password, follow the instructions.

> **Note:** If you specify this password, it is used and the password set in the `realm.properties` file for the user (if that user exists in the file) is ignored.

d. Save the file.

5. The default authentication mechanism is set to Basic. To change the authentication to Digest,
   a. Open the `BW_HOME\config\bwagent.ini` file in a text editor.
   b. Navigate to the **Web server HTTP and HTTPS configuration** section of the file and uncomment the **bw.agent.bw.auth** property. Change the value from **BASIC** to **DIGEST** (all caps).
   c. Add a new property called **bw.agent.bw.api.auth**. Set the value of this property to **DIGEST** (all caps).
   d. Save the file.
6. Set the **bw.agent.http.authorization** property to **true** in the `bwagent.ini` file for authentication and authorization.
7. Restart the BWAgent.

Enabling LDAP Authentication for the BWAgent REST API

To enable LDAP authentication for the BWAgent REST API, perform the following steps:

Before you begin

- Stop the BWAgent.
- Ensure that users are assigned the admin, operator, and user roles in the LDAP server. For more information, see Authorizing Access to the REST API by Role.

Procedure

1. Navigate to `BW_HOME\bin`.
2. Open the `bwagent.tra` file in an editor and uncomment the following property:

```
java.property.java.security.auth.login.config=%BW_
HOME%/config/jaas.login.conf
```

3. Replace the following lines to the `%BW_HOME%/config/jaas.login.conf` file.

```
bwloginmodule {
  org.eclipse.jetty.jaas.spi.LdapLoginModule required
  debug="true"
  useLdaps="false"
  contextFactory="com.sun.jndi.ldap.LdapCtxFactory"
  hostname="10.97.106.72"
  port="10389"
  authenticationMethod="simple"
  forceBindingLogin="true"
  userBaseDn="o=tibco"
  userObjectClass="inetOrgPerson"
  userRdnAttribute="uid"
  userIdAttribute="uid"
  userPasswordAttribute="userPassword"
  roleBaseDn="ou=roles,o=tibco"
  roleNameAttribute="cn"
  roleMemberAttribute="uniqueMember"
  customRoleForLDAP=test
  roleObjectClass="groupOfUniqueNames";
};
```

4. In the `bwagent.ini` file, set the **bw.agent.http.authorization** property to **true** for LDAP authentication and authorization, and set the property to **false** to enable authorization for the Custom Group.
5. Restart the BWAgent.

6. Open a web browser and go to the URL of the API server. For example, enter the following URL:

   `http://localhost:8079/bw/v1/agents/info`

7. Enter the correct LDAP username and password credentials in the dialog that displays.

Enabling LDAP Over SSL Authentication for the BWAgent REST API

To enable LDAP over SSL authentication for the BWAgent REST API, perform the following steps:

Before you begin

- Stop the BWAgent.
- Ensure that users are assigned the admin, operator, and user roles in the LDAP server. For more information, see Authorizing Access to the REST API by Role.

Procedure

1. Navigate to `BW_HOME\bin`.
2. Open the `bwagent.tra` file in an editor and uncomment the following property:

   ```
   java.property.java.security.auth.login.config=%BW_
   HOME%/config/jaas.login.conf
   ```

3. Replace the following lines to the `%BW_HOME%/config/jaas.login.conf` file.

   ```
   bwloginmodule {
     org.eclipse.jetty.jaas.spi.LdapLoginModule required
     debug="true"
     useLdaps="true"
     contextFactory="com.sun.jndi.ldap.LdapCtxFactory"
     hostname="localhost"
     port="10636"
     bindDn="cn=John Keats,ou=users,o=mojo"
     bindPassword="pass"
   ```

```
  authenticationMethod="simple"
  forceBindingLogin="true"
  userBaseDn="o=mojo"
  userObjectClass="inetOrgPerson"
  userRdnAttribute="uid"
  userIdAttribute="uid"
  userPasswordAttribute="userPassword"
  roleBaseDn="ou=bwgroups,o=mojo"
  roleNameAttribute="cn"
  roleMemberAttribute="uniqueMember"
  roleObjectClass="groupOfUniqueNames";
};
```

Note: Ensure that the value of the attribute useLdaps is set to **true**.

4. For more information about importing the LDAP SSL certificate into the cacerts keystore file that is shipped with `tibcojre`, see Importing the LDAP SSL Certificate in the Cacerts Keystore File.
5. Restart the BWAgent.
6. Open a web browser and go to the URL of the API server. For example, enter the following URL:

   `http://localhost:8079/bw/v1/agents/info`
7. Enter the LDAP username and password credentials.

Authorizing Access to the REST API by Role

The REST API supports the following roles: admin, operator, and user. Roles are assigned in the `realm.properties` file. Multiple roles can be assigned to a single user.

Before you begin

Stop the BWAgent if it is running.

Role	Description	Access to Operations
admin	Full rights: Create, Read, Update, Delete, Lifecycle (default)	All
operator	Read and Lifecycle operations	Start/Stop, GET
user	Read only	GET

Procedure

1. Navigate to `BW_HOME\config` and open the **realm.properties** file. For more information, see Enabling Authentication for the BWAgent REST API.
2. For each defined user, change the default role as needed.
3. Save the file and restart the BWAgent.

Securing the REST API Server

The BWAgent REST API server can be secured with an SSL connection.

The SSL connection is configured in the BWAgent configuration file.

Before you begin

Stop the BWAgent if it is running.

Procedure

1. Open the `BW_HOME\config\bwagent.ini` file in a text editor and navigate to **Web server HTTP and HTTPS configuration**.
2. Comment out the **bw.agent.http.port** property.
3. Uncomment the **bw.agent.https.port** property.

   > **Note:** If the `bw.agent.http.port` property is also set, requests to the HTTP port are automatically rerouted to the HTTPS port.

4. Uncomment the following properties and add values:

Property	Value	Description
bw.agent.https.truststorepath	String	Path to the trust store file.
bw.agent.https.truststorepassword	String	The password for the trust store.
bw.agent.https.keystorepath	String	Path to the keystore file.
bw.agent.https.keystorepassword	String	Password of the keystore.

Note: If you are using a self-signing certificate, specify **CN** to be `localhost`.

5. Add additional SSL properties to the **Web server HTTP and HTTPS configuration** section of the `bwagent.ini` file as needed.

Property	Value	Description
bw.agent.https.allowrenegotiate	true|false	Set if SSL renegotiation is allowed.
bw.agent.https.certalias	String	Alias of SSL certificate for the connector.
bw.agent.https.keymanagerpassword	Obfuscated password	The password (if any) for the specific key in the keystore.
bw.agent.https.keystoretype	String	The type of the keystore (default "JKS").
bw.agent.https.needclientauth	true|false	Set to **true** if SSL needs client authentication.
bw.agent.https.protocol	String	The SSL protocol (default "TLS") passed as the

Property	Value	Description
		protocol parameter to the `SSLContext.getInstance (String, String)` method.
`bw.agent.https.provider`	String	The SSL provider name, which, if set, is passed as the provider parameter to the `SSLContext.getInstance (String, String)` method.
`bw.agent.https.trustall`	true|false	Set to **true** if all certificates should be trusted (for example, if there is no Keystore or TrustStore).
`bw.agent.https.trustmanagerfactoryalgorithm`	String	The algorithm name (default "SunX509") passed to `TrustManagerFactory`. Use the string "TrustAll" to install a trust manager that trusts all.
`bw.agent.https.truststoreprovider`	String	The provider of the trust store.
`bw.agent.https.truststoretype`	String	The type of the trust store (default "JKS").
`bw.agent.https.ocspresponderurl`	String	Location of the OCSP Responder.

Property	Value	Description
bw.agent.https.enablecrldp	true|false	Set to **true** to enable CRL Distribution Points support.
bw.agent.https.enableocsp	true|false	Set to **true** to enable On-Line Certificate Status Protocol support.
bw.agent.https.crlpath	String	The path to the file that contains the Certificate Revocation List.
bw.agent.https.validatecerts	true|false	Set to **true** if SSL certificates have to be validated.
bw.agent.https.validatepeercerts	true|false	Set to **true** if the SSL certificates of the peer have to be validated.
bw.agent.https.wantclientauth	true|false	Set to **true** if SSL wants client authentication.
bw.agent.https.maxcertpathlength	int	The maximum number of intermediate certificates in the certification path (-1 for unlimited).
bw.agent.https.excludeciphersuites	Comma-separated list of Strings	The list of cipher suite names to exclude from.
bw.agent.https.includeciphersuites	Comma-separated	The list of cipher suite names to include.

Property	Value	Description
	list of Strings	
`bw.agent.https.securerandomalgorithm`	String	The algorithm name. If set, is passed as the **algorithm** parameter to `SecureRandom.getInstance(String)` method to obtain the **SecureRandom** instance, which is then passed to the SSLContext constructor.
`bw.agent.https.excludeprotocols`	String	Add this property to disable SSL protocols. Disabled SSL protocols are represented in a comma-delimited list.
`bw.agent.https.includeprotocols`	String	Add this property to enable SSL protocols. Enabled SSL protocols are represented in a comma-delimited list.

6. Save the file and restart the BWAgent.
7. Open a web browser and go to the URL of the API server. For example, enter the following URL:

   `https://localhost:8886/bw/v1/agents/info`

Importing the LDAP SSL Certificate in the Cacerts Keystore File

To connect the BWAgent to the LDAP Over SSL server, ensure that the server certificate is imported into the cacerts keystore file.

Before you begin

Ensure that the SSL Certificate has been exported from the LDAP server.

Procedure

1. Navigate to `BW_Home\tibcojre64\1.8.0\lib\security`

```
# To List Existing Certificates use command:

BW_Home\tibcojre64\1.8.0\lib\security>keytool -list -keystore
cacerts
Enter keystore password: changeit


Keystore type: JKS
Keystore provider: SUN


Your keystore contains 5 entries


verisignclass2g2ca [jdk], Aug 25, 2016, trustedCertEntry,
Certificate fingerprint (SHA1):
B3:EA:C4:47:76:C9:C8:1C:EA:F2:9D:95:B6:CC:A0:08:1B:67:EC:9D
digicertassuredidg3 [jdk], Aug 25, 2016, trustedCertEntry,
Certificate fingerprint (SHA1):
F5:17:A2:4F:9A:48:C6:C9:F8:A2:00:26:9F:DC:0F:48:2C:AB:30:89
verisignuniversalrootca [jdk], Aug 25, 2016, trustedCertEntry,
Certificate fingerprint (SHA1):
36:79:CA:35:66:87:72:30:4D:30:A5:FB:87:3B:0F:A7:7B:B7:0D:54
digicerttrustedrootg4 [jdk], Aug 25, 2016, trustedCertEntry,
Certificate fingerprint (SHA1):
DD:FB:16:CD:49:31:C9:73:A2:03:7D:3F:C8:3A:4D:7D:77:5D:05:E4
verisignclass1g3ca [jdk], Aug 25, 2016, trustedCertEntry,
Certificate fingerprint (SHA1):
20:42:85:DC:F7:EB:76:41:95:57:8E:13:6B:D4:B7:D1:E9:8E:46:A5
identrustpublicca [jdk], Aug 25, 2016, trustedCertEntry,
E:\BW\6.4.0\V11\tibcojre64\1.8.0\lib\security>
```

```
#### Import LDAP SSL Certificates to cacerts:

BW_Home\tibcojre64\1.8.0\lib\security>keytool -import -keystore
cacerts -file <certName>.der

Enter keystore password: changeit

Owner: CN=bwin2k8r264b_52, OU=Directory, O=ASF, C=US

Issuer: CN=ApacheDS, OU=Directory, O=ASF, C=US

Serial number: 15ac649f77e

Valid from: Sun Mar 12 23:10:20 PDT 2017 until: Mon Mar 12 23:10:20
PDT 2018

Certificate fingerprints:

      MD5:  A4:25:84:6C:63:51:C5:A2:EB:D5:69:2A:74:EE:D3:31

      SHA1:
F0:9D:0A:26:E3:86:61:CB:62:3F:1F:40:5A:31:F3:BC:0C:C9:C0:B0

      SHA256:
82:43:35:95:55:A6:CC:36:BB:C8:9A:6E:9D:55:FF:69:C1:7C:30:B3:EC:79:D
A:3E:98:A9:F2:B6:5C:48:B8:28

      Signature algorithm name: SHA1withRSA

      Version: 1

Trust this certificate? [no]:  yes

Certificate was added to keystore


#### Check if the certificate was imported. The number of keystore
entries should increase by 1.


BW_Home\tibcojre64\1.8.0\lib\security>keytool -list -keystore
cacerts

Enter keystore password:


Keystore type: JKS

Keystore provider: SUN

Your keystore contains 6 entries
```

```
verisignclass2g2ca [jdk], Aug 25, 2016, trustedCertEntry,

Certificate fingerprint (SHA1):
B3:EA:C4:47:76:C9:C8:1C:EA:F2:9D:95:B6:CC:A0:08:1B:67:EC:9D

digicertassuredidg3 [jdk], Aug 25, 2016, trustedCertEntry,

Certificate fingerprint (SHA1):
F5:17:A2:4F:9A:48:C6:C9:F8:A2:00:26:9F:DC:0F:48:2C:AB:30:89

verisignuniversalrootca [jdk], Aug 25, 2016, trustedCertEntry,

Certificate fingerprint (SHA1):
36:79:CA:35:66:87:72:30:4D:30:A5:FB:87:3B:0F:A7:7B:B7:0D:54

digicerttrustedrootg4 [jdk], Aug 25, 2016, trustedCertEntry,

Certificate fingerprint (SHA1):
DD:FB:16:CD:49:31:C9:73:A2:03:7D:3F:C8:3A:4D:7D:77:5D:05:E4

verisignclass1g3ca [jdk], Aug 25, 2016, trustedCertEntry,

Certificate fingerprint (SHA1):
20:42:85:DC:F7:EB:76:41:95:57:8E:13:6B:D4:B7:D1:E9:8E:46:A5

identrustpublicca [jdk], Aug 25, 2016, trustedCertEntry,

Certificate fingerprint (SHA1):
BA:29:41:60:77:98:3F:F4:F3:EF:F2:31:05:3B:2E:EA:6D:4D:45:FD

utnuserfirstobjectca [jdk], Aug 25, 2016, trustedCertEntry,

BW_Home\tibcojre64\1.8.0\lib\security>
```

Viewing BWAgent Information

Use the BWAdmin `show` command to view information about the installation, including BWAgent name, process ID, number of active threads, memory and CPU usage, and up time.

BWAdmin Command Line

Make sure that the BWAgent is running, then issue the following command, providing the agent name in the command line.

```
BW_HOME\bin>bwadmin show agentprocessinfo M1
```

The following details are returned:

```
Agent Name:                    M1
System Process ID:             20976
Number of Active Threads:      163
Total Memory(in bytes):        1174147072
Free Memory(in bytes):         834506752
Used Memory(in bytes):         339640320
Percent Memory Used:           28.0
Percent CPU Used:              0.0
Up Time:                       0d 00:16:40
```

Restoring the File System of a BWAgent

Restoring a BWAgent restores the file system of the BWAgent to the state of the datastore. The file system for runtime entities in a BWAgent can be restored locally or across a network, if the BWAgent is part of an agent network.

Before you begin

- The name of the BWAgent must be known in order to restore.
- The BWAgent must be running.

Procedure

1. To restore the file system for runtime entities in a BWAgent, open a terminal and navigate to *BW_HOME*\bin.
2. Enter the `restore` command at the command line, using the `agent` argument with the name of the BWAgent to restore. The BWAgent can be either the local BWAgent or a BWAgent in the agent network. The following example restores the BWAgent named Machine1.

```
BW_HOME\bin>bwadmin restore agent Machine1
```

3. To verify the restore, check the file system. Open the *BW_HOME*\domains folder. Look for named domain folders to verify if domains for the BWAgent have been restored.

Configuring the Location of the Domains Folder

Runtime entities are created in the local file system in the *BW_HOME*/domains folder. Below are several different ways to change the folder location of a specific domain.

Admin CLI

Point to a new domain home for a specific domain by running the -home command. For example, run the following command to create a domain home named testDomain:

```
<BW_HOME>\ bin>bwadmin create -home /Users/testuser/domains domain
testDomain
```

REST API

Use the home parameter in the REST request. For example, http://localhost:8079/bw/v1/domains/testDomain?home=/Users/testuser/domains

Domain Home Properties File

Update the domain home properties file to change the folder location of a specific domain.

Procedure

1. Stop the BWAgent if it is running.
2. Open the *BW_HOME*/domains/DomainHomes.properties file in a text editor.
3. Add the <*domainName*>.domainHome property, where <*domainName*> is the intended domain, to this file to point to the new domain home for a specific domain. For example, by adding the testDomain.domainHome property to the DomainHomes file, you are specifying a custom domain folder for the domain testDomain.

4. Save the file and restart the BWAgent.
5. From the Admin UI, create the domain using the domain name you specified in the `domainHome` property. For example, if you have added `testDomain.domainHome` to the DomainHomes file, go to the Admin UI and create a domain called `testDomain`.

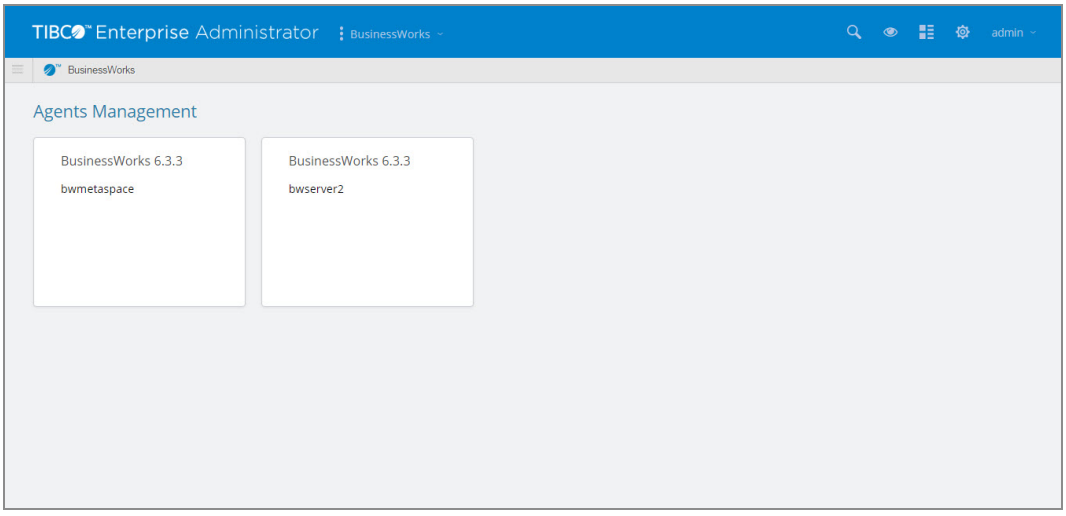
Using BWAgent with TEA

The Admin UI is a web UI that runs in TIBCO® Enterprise Administrator (TEA). To enable the Admin UI, the BWAgent must be registered with a running TEA server. Use the Admin UI to create, view, and monitor runtime entities.

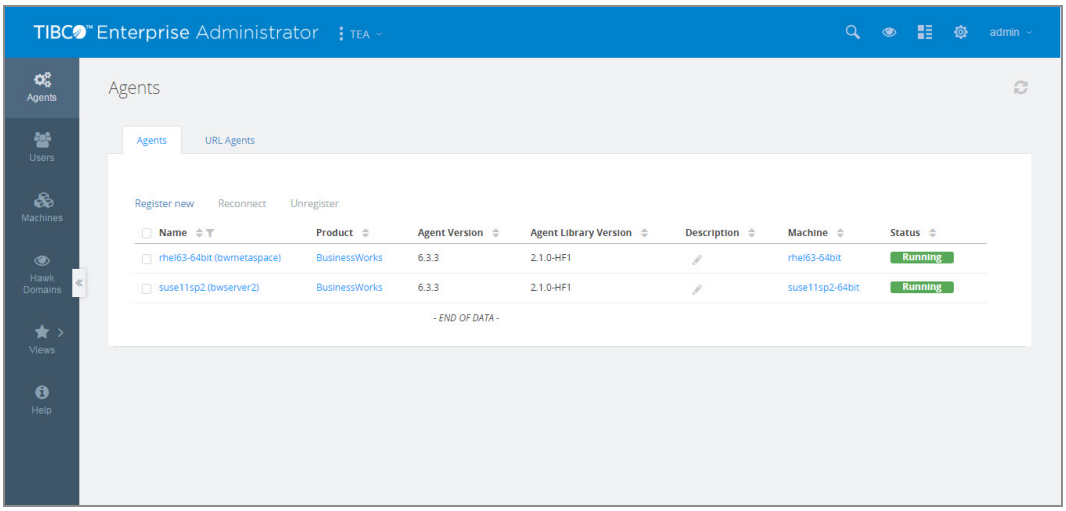
The BWAgent interacts with the TIBCO Enterprise Administrator server through a TEA agent. Multiple BWAgents can be registered with TEA, but only one TEA agent can be registered with a BWAgent at a time.

Note: If you register multiple BWAgents with a TEA agent, ensure that the BWAgents are using the same version of ActiveMatrix BusinessWorks 6.x.

Admin UI



Once the TEA agent is registered with a BWAgent, the BWAgent is displayed in the Admin UI, and the Admin UI can be used to manage and monitor runtime entities.



The Admin UI can be used to perform almost all BWAdmin administrative tasks. For a walk-through of the steps to start working with the Admin UI, see Running Applications in Enterprise Mode using the Admin UI.

For an agent network to be managed from the Admin UI, one BWAgent in the agent network must be registered with the TIBCO Enterprise Administrator server. If the registered BWAgent ends, the connection between the server and the agent network is automatically recovered. Another BWAgent in the agent network is automatically registered with the server.

Registering BWAgent with TIBCO Enterprise Administrator

A BWAgent TEA agent must be registered with the TIBCO Enterprise Administrator server before it is available in the Admin UI.

BWAdmin Command Line

Procedure

1. Start the TIBCO Enterprise Administrator server.
2. Start BWAgent.
3. Register the BWAgent TEA agent with the TIBCO Enterprise Administrator server.

Provide the URL to the TEA server. Only one TEA agent can be registered at a time.

```
BW_HOME\bin>bwadmin registerteaagent http://<TEA_HOST>:8777/tea/
```

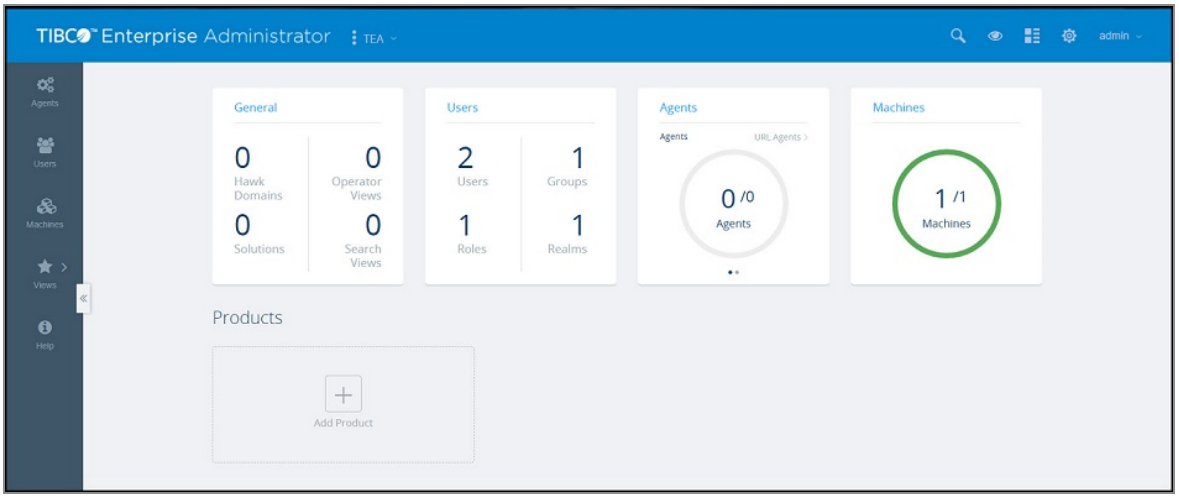
For more information about creating and managing runtime entities, see Administration Tasks and Reference.

Admin UI

Procedure

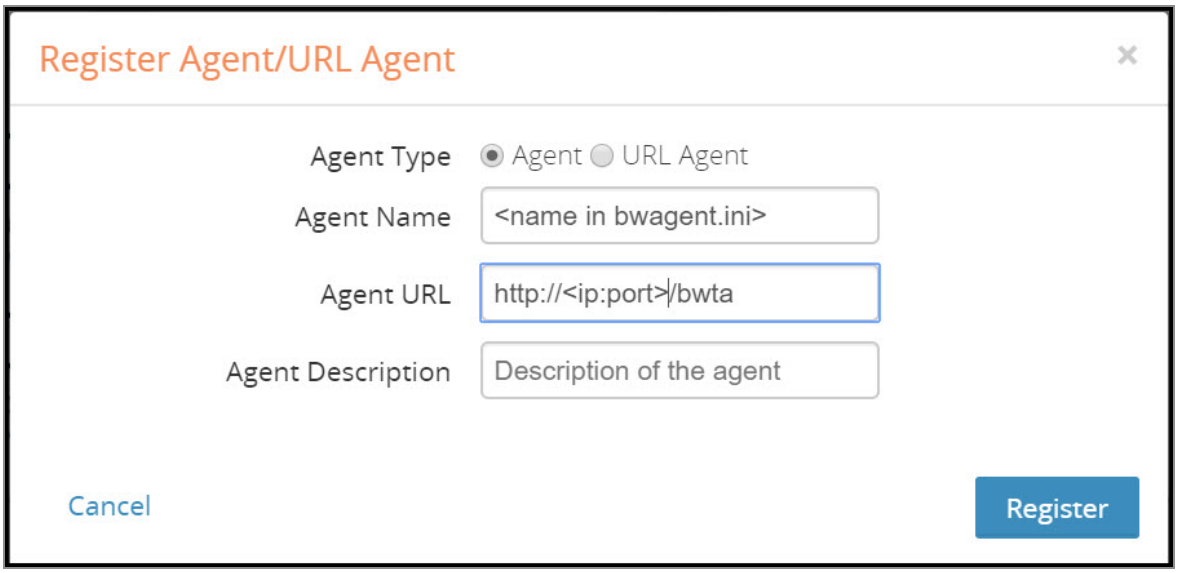
1. Open Admin UI (`http://<TEA_HOST>:8777/tea`) to access the BWAgent. Login credentials are required. The default username is admin and the default password is admin.

   Admin UI Home Page is displayed.



2. Click the **Add Product** icon on the Admin UI home page.

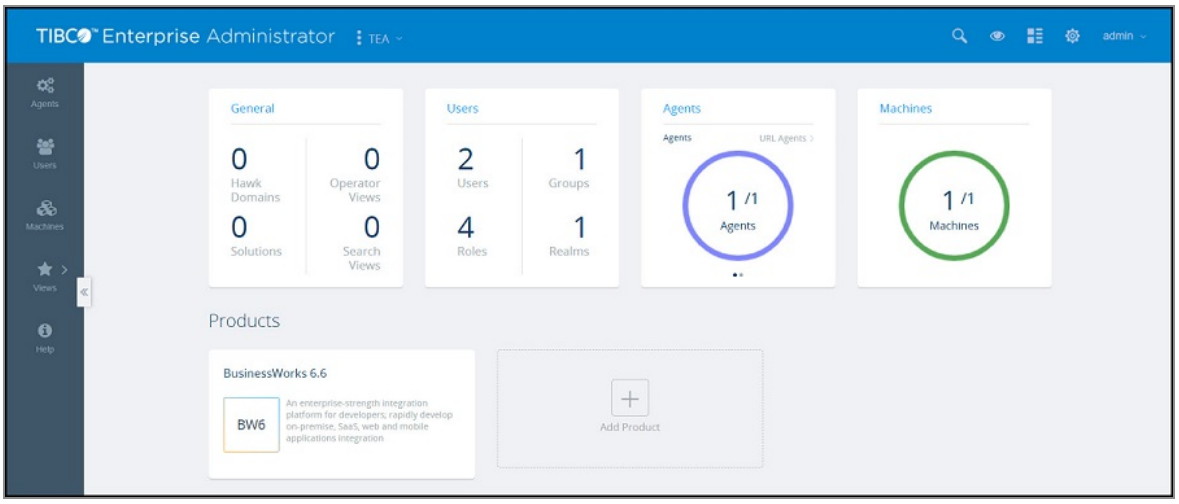
   The Register Agent/URL Agent dialog is displayed.



3. Add the following information:

Field	Value
Agent Name	Name of the BWAgent configured in the `bwagent.ini` file.
Agent URL	URL should be in the `http://<ip:port>/bwta` format. The default port is 9091.
Agent Description	Optional. Description of the BWAgent.

4. Click **Register**.



Autoregistering BWAgent with TIBCO Enterprise Administrator

For an agent network to be managed from the Admin UI, one BWAgent in the agent network must be registered with the TIBCO Enterprise Administrator (TEA) server. If the registered BWAgent ends, the connection between the server and the agent network is automatically recovered. Another BWAgent in the agent network is automatically registered with the server.

The BWAdmin `disableautoregistration` and `enableautoregistration` commands toggle the mechanism used to autoregister a BWAgent. The commands can be run against any BWAgent in an agent network and act on all BWAgents in the agent network. If you disable autoregistration for an agent network, the members of the network are unable to communicate with the TIBCO Enterprise Administrator server. You have to register a BWAgent in the agent network manually to communicate it with the server.

If the same TEA server is used for different versions of BWAgents and only one version of BWAgent is up, follow these steps to make sure TEA is working with appropriate UIs:

Procedure

1. Stop the TEA server.
2. Set the property `tea.dev.developer-mode` to true in the `<TEA-HOME>/tibco/cfgmgmt/tea/conf/tea.conf` file.

3. Restart the TEA server.
4. As an admin user, on the Agents page, click the **Reload** button.
5. Verify if new UI changes are picked up.
6. Stop the TEA server.
7. Reset `tea.dev.developer-mode` to false.
8. Restart the TEA server.
9. Clear the cache and reload Admin UI.

Enabling and Disabling BWAgent's TIBCO Enterprise Administrator Agent Port

You can disable BWAgent's TEA agent port to disable registering BWAgent with TIBCO Enterprise Administrator.

Run the `bwagent.exe startagent -nt` command from the Admin Console. While running the command, you are not able to register BWAgent with Admin UI.

To enable the registering again, restart BWAgent with the `bwagent.exe` command without the `startagent -nt` option.

Unregistering BWAgent with TIBCO Enterprise Administrator

You can unregister BWAgent from the Admin UI, or the command line.

Unregistering BWAgent Using the Admin UI

To unregister BWAgent from the Admin UI, open the TEA URL and perform these steps:

1. Click **Agents** on the side bar to open the Agent page.
2. Select BWAgent to unregister.

3. Click **Unregister**.
4. In the dialog window that displays, confirm that you want to unregister BWAgent.

BWAgent is no longer registered with the TEA server.

Unregistering BWAgent Using the Command Line

Use BWAdmin to run the `unregisterteaagent` command, and enter the URL of your TEA server:

```
BW_HOME\bin>bwadmin unregisterteaagent <TEA URL>
```

Compatibility Chart for TIBCO ActiveMatrix BusinessWorks™ and TIBCO® Enterprise Administrator

The TIBCO Enterprise Administrator (TEA) server is an application administration UI that supports multiple TIBCO products, including the ActiveMatrix BusinessWorks. Using the Admin UI you can create, view, and monitor runtime entities. Each product registers its own agent with the server and the server communicates with the products through these agents. The compatibility rules and chart help determine the minimum version of the TEA server required by a given version of ActiveMatrix BusinessWorks.

These are the compatibility rules for the TEA server and TEA agent libraries:

- The TEA server is backward compatible with earlier versions of TEA agent libraries, unless a known issue exists.
- The TEA server does not guarantee forward compatibility with newer versions of TEA agent libraries.

The following table lists the version of the TEA agent library bundled with a given version of ActiveMatrix BusinessWorks.

ActiveMatrix BusinessWorks Version	Version of TEA Agent Library Bundled
6.7.0	2.4.0 HF-001
6.8.0	2.4.0 HF-006
6.8.1	2.4.1
6.9.0	2.4.1
6.9.1	2.4.1 HF-002
6.10.0	2.4.1
6.11.0	2.4.2

Compatibility Chart

Based on compatibility rules and the version of the TEA agent library bundled in a given version of ActiveMatrix BusinessWorks, see the following compatibility chart:

ActiveMatrix BusinessWorks Version with TEA Agent Library Version	TEA Server Version			
	2.4.0	2.4.1	2.4.1 HF-002	2.4.2
ActiveMatrix BusinessWorks 6.7.0 with TEA Agent Library 2.4.0 HF-001	✓	✗	✗	✗
ActiveMatrix BusinessWorks 6.8.0 with TEA Agent Library 2.4.0 HF-006	✓	✗	✗	✗
ActiveMatrix BusinessWorks 6.8.1 with TEA Agent Library 2.4.1	✗	✓	✗	✗
ActiveMatrix BusinessWorks 6.9.0 with TEA Agent Library 2.4.1	✗	✓	✗	✗

ActiveMatrix BusinessWorks Version with TEA Agent Library Version	TEA Server Version			
	2.4.0	2.4.1	2.4.1 HF-002	2.4.2
ActiveMatrix BusinessWorks 6.9.1 with TEA Agent Library 2.4.1 HF-002	✗	✓	✓	✗
ActiveMatrix BusinessWorks 6.10.0 with TEA Agent Library 2.4.1	✗	✓	✓	✗
ActiveMatrix BusinessWorks 6.11.0 with TEA Agent Library 2.4.2	✗	✓	✓	✓

TEA Shell

A command line utility called the TEA shell is provided with TIBCO Enterprise Administrator server. It is a remote shell based on the SSH protocol that provides the command line interface for the full range of TEA operations. The scripting language is similar to that of bash from Unix.

The TEA shell has the following key features:

- Piping of commands
- Completion of commands
- Help on commands

The BusinessWorks entity structure in the TEA shell is:

```
/BusinessWorks
bwagents
domains
   apparchives
   appspaces
     applications
     appnodes
installations
machines
```

```
    bwagents
    installations
```

To switch to the BusinessWorks context, type `admin@M1:/> cd Businessworks`

Press the tab key for a list of available commands for the context path.

For information about TEA shell commands, see Using TEA Shell Commands.

Using TEA Shell Commands

TEA shell commands can be used to create, monitor, and manage runtime entities.

TEA shell commands are aligned with BWAdmin commands.

The steps in this section show you some simple TEA shell commands for creating a domain, AppSpace, and AppNode and starting the AppSpace. For a complete list of all supported commands, see TEA Shell Commands.

At any time in the TEA shell, press the tab key for a list of supported commands available for the context. To get help on a command, type the command with the --help option, for example: `create --help`

Procedure

1. Connect to the TEA shell through a terminal program, for example Putty. Connect using the following command:

   ```
   ssh -p 2222 admin@localhost
   ```

   The username and password are both `admin`

   On successful connection, the following TEA Shell banner is displayed:

2. To switch to the BusinessWorks context, type `admin@M1:/> cd Businessworks`
3. Create a domain:

```
admin@M1:/BusinessWorks> create domain TEA-D1
Executed the command 'create' successfully.
```

4. Create an AppSpace in the domain:

```
admin@M1:/BusinessWorks> create -domain TEA-D1 appspace TEA-AS1
Executed the command 'create' successfully.
```

5. Create an AppNode in the AppSpace:

```
admin@M1:/BusinessWorks> create -domain TEA-D1 -appspace TEA-AS1
appnode TEA-AN1 -httpPort 8077
Executed the command 'create' successfully.
```

6. Start the AppSpace. This starts the AppNode in the AppSpace.

```
admin@M1:/BusinessWorks> start -domain TEA-D1 appspace TEA-AS1
Executed the command 'start' successfully.
```

TEA Shell Commands

This topic lists all TEA shell commands and provides examples.

General Commands

Command	Description	Example
cd	Changes context to entity.	`cd /BusinessWorks/domains/Domain1/appspaces/AppSpace01/appnodes/AppNode01`
ls	Lists the name of each instance in the specified entity.	`cd /BusinessWorks/domains/Domain1/appspaces/AppSpace01/appnodes` `ls`

Domain Commands

Command	Example
Create domain	`cd /BusinessWorks/domains` `create -domain Domain1 -descr "Sanity Test Domain"`
Delete domain	`cd /BusinessWorks/domains` `delete -domain Domain1`

AppSpace Commands

Command	Example
Create an AppSpace. The `-minNodes` parameter is optional and defaults to 1	`cd /BusinessWorks/domains/Domain1/appspaces` `create -appspace AppSpace01 -descr"AppSpace 01"-minNodes"2"`

Command	Example
Delete an AppSpace	`cd /BusinessWorks/domains/Domain1/appspaces delete -appspace AppSpace01`
Start an AppSpace	`cd /BusinessWorks/domains/Domain1/appspaces start -appspace AppSpace01`
Stop an AppSpace	`cd /BusinessWorks/domains/Domain1/appspaces stop -appspace AppSpace01`

AppNode Commands

Description	Example
Create an AppNode; `-osgiPort` parameter is optional.	`cd /BusinessWorks/domains/Domain1/appspaces/AppSpace01/appnodes create -appnode Sanity-AppNode01 -httpPort 7011 -agent localhost -osgiPort 8011`
Delete an AppNode	`cd /BusinessWorks/domains/Domain1/appspaces/AppSpace01/appnodes delete -appnode Sanity-AppNode01`
Start an AppNode	`cd /BusinessWorks/domains/Domain1/appspaces/AppSpace01/appnodes start -appnode Sanity-AppNode01`
Stop an AppNode	`cd /BusinessWorks/domains/Domain1/appspaces/AppSpace01/appnodes stop -appnode Sanity-AppNode01`

Application Commands

Description	Example
Start an application	`cd /BusinessWorks/domains/Domain1/appspaces/AppSpace01/applications` `start -app acme.acct.ap.application -version 1.0`

Start an application instance on the AppNode	`cd`
	
	`/BusinessWorks/domains/Domain1/appspaces/AppSpace01/applications`
	`startinst -app acme.acct.ap.application -version1.0-appnode Sanity-AppNode01`

Stop an application	`cd /BusinessWorks/domains/Domain1/appspaces/AppSpace01/applications` `stop -app acme.acct.ap.application -version 1.0`
Stop an application instance on an AppNode	`d /BusinessWorks/domains/Domain1/appspaces/AppSpace01/applications` `stopinst -app acme.acct.ap.application -version 1.0 -appnode Sanity-AppNode01`

Roles and Permissions

Privileges to perform actions and operations in the Admin UI are based on the role and permissions granted to the user.

Roles

Administrators use roles to allot permissions in the Admin UI. When a role is assigned to a user or a group, the user or group receives all the permissions granted to the role.

The following roles are defined in the Admin UI: BW User, BW Operator, and BW Administrator.

Permissions

Permissions are used to enforce access control. In the Admin UI, you can grant access permissions at two levels:

- Entity Based - Permission can be enforced on the complete entity such as a domain, or an AppSpace. For example, for two domains d1 and d2, if the read permission is granted to the domain entity, you can view both the instances, d1 and d2.
- Instance Based - Permission can be enforced on a particular instance of an entity. For example, if the read permission is granted for the d1 instance of the domain entity, you have permission to view only d1.

Types of Permissions

You can assign the following types of permissions to users in the Admin UI:

- Read: Read permission for the entities.
- Lifecycle: You can grant lifecycle permission to a user only if he has explicit read permission. Lifecycle permission is applicable only to AppSpaces, AppNodes, and applications, to control the lifecycle of entities such as AppSpaces, AppNodes, and applications (that is to start and stop the entities)
- Full_control: By default, this permission includes the read permission. Entities can perform the following commands with the full control access:

- Domain: delete and back up
- Domains: create
- Archive: deploy and delete
- Archives: upload
- AppSpace: delete, create AppNode and update
- AppSpaces: create
- AppNode: delete and update
- AppNodes: create
- Application: update and undeploy
- Agent: unregister

Role-Based Permissions

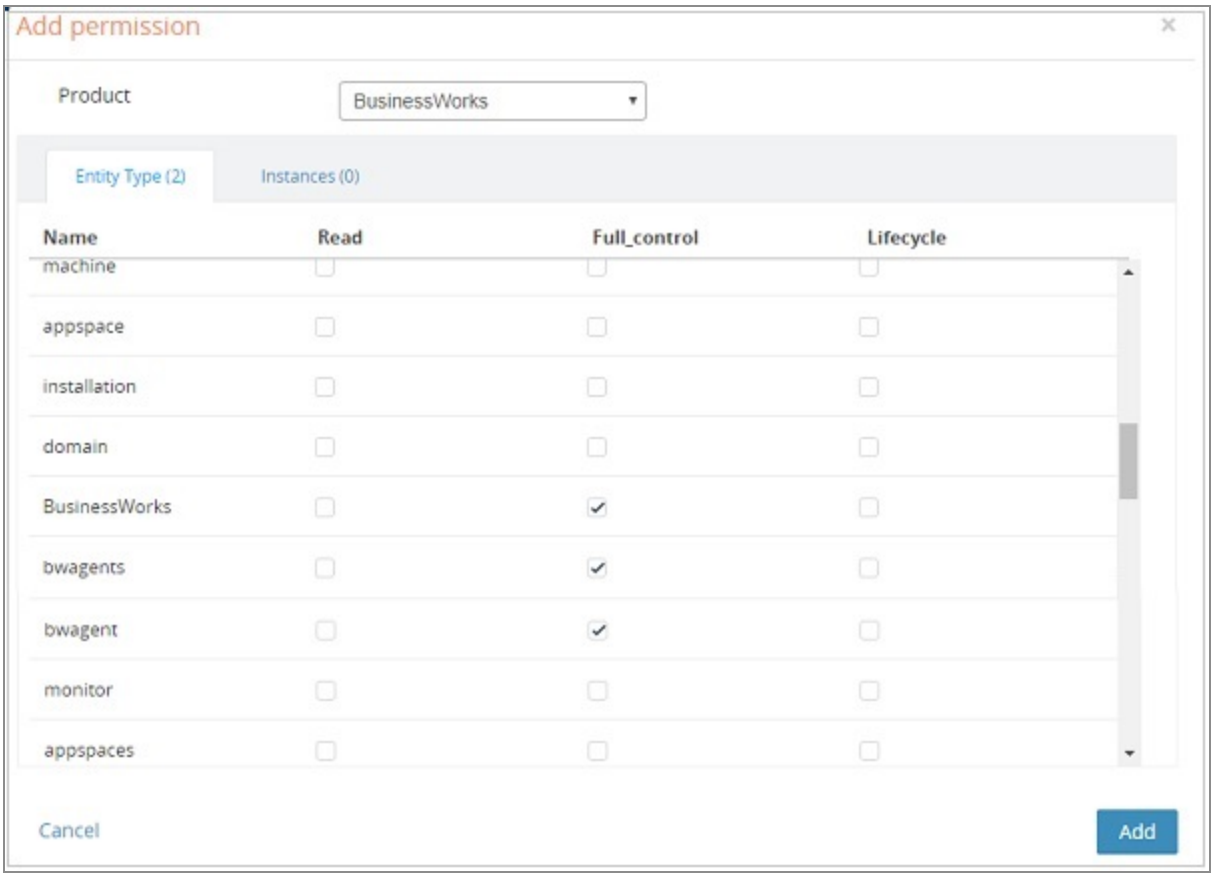
The following table shows permissions for the entities:

Roles	Permissions
BW User	Read
BW Operator	Lifecycle
BW Administrator	Full_control

Entity-Based Permissions

To add the ActiveMatrix BusinessWorks product under the list of **Products** on the Admin UI homepage, select the following entities on the **Add Permission** page.

- **BusinessWorks**
- **bwagent**
- **bwagents**



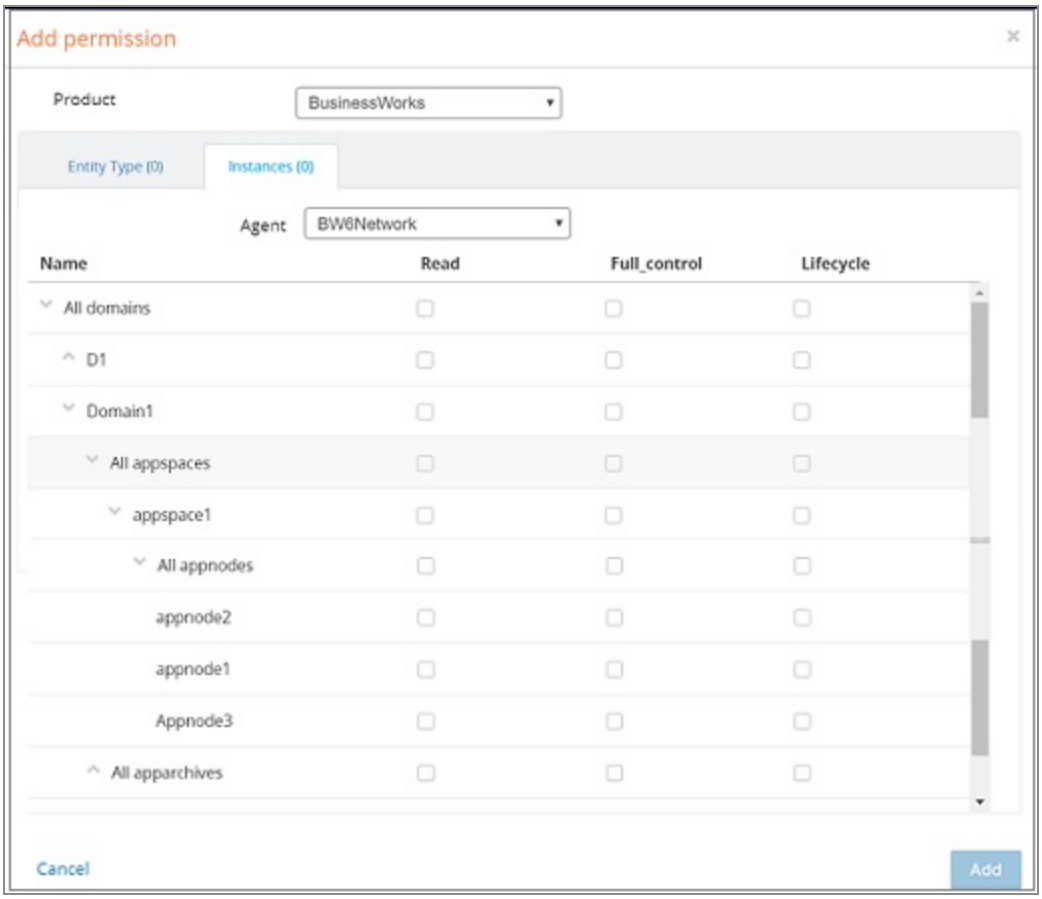
To manage permissions for an application, select the application and applications entities on the **Add Permission** page and assign the required permissions.

Note: Grant access permissions to plural entities to access an instance of the entity. For example, to give read access to **d1**, which is an instance of a domain, grant the read permission to the domains (plural entity). This is applicable to all entity types.

Entity Hierarchy for Instance-Based Permission

The hierarchy of entities when granting permissions in Admin UI is illustrated in the following image. Domain is the top-level parent entity and includes AppSpaces and Archives. AppSpaces then further include AppNodes.

Note: If an archive is uploaded to a folder, provide access for the folder first and then to the archive instances.



Granting Instance-Based Permissions

Scenario 1: Instance-based permission assigned to a child entity

When an instance-based permission is assigned to a child entity, the read permission is assigned to the parent entity if the parent does not have any permissions assigned. The administrator can, however, update the permission assigned to the parent. The updated permission is then enforced.

Scenario 2: Instance-based permission assigned to a parent entity

When an instance-based permission is assigned to a parent entity, the permission is not applied to the child entity. Permissions for the child entities can be assigned explicitly. For example, if the read permission is applied to AppSpace1, the child entities of AppSpace1 do not inherit the permission.

Example of how entity-based and instance-based permissions work

Objective

AppSpace a1 contains two AppNodes, n1 and n2. AppSpace a1 is a child entity of Domain d1. Grant permissions so that you can only view AppNode n1 and start and stop AppNode n2.

1. From the entity permission page, **Add Permission**, provide read permissions to the entities BusinessWorks and BWAgents. They are the top-level entities and are mandatory to view the TIBCO ActiveMatrix BusinessWorks™ product.
2. Provide entity-based permission to domains, AppSpaces, and AppNodes. It is mandatory to provide permission for plural entities such as domains and AppSpaces, to view the content on these pages.
3. Provide instance-based permission to AppNode n1 and lifecycle and read permission to AppNode n2.

The following section explains how the permissions granted in the example work:

- For Domain d1 to be visible, grant permissions to the entities: BusinessWorks, BWAgents, domains and for the instance of the domain d1.
- For AppSpace a1 to be visible, grant permission to the entities: BusinessWorks, BWAgents, domains, AppSpaces and for the instance a1. Explicit permissions are not required for Domain d1. Parents entities are provided view permission automatically.
- For AppNode n1 to be visible, grant permission to the entities: BusinessWorks, BWAgents and domains, AppSpaces and AppNodes and for the instance a1.

Additional Notes

- Actions taken on the parent level transcend the actions taken on the child entity even if you do not have access to the child entity. For example, if you start and stop an AppSpace, all the AppNodes in this AppSpace start and stop even if you do not have access to all the AppNodes.
- Custom users cannot view any new entity they create as these users do not have instance-based permission for that entity. For example, you have full control access to an AppSpace and you navigate to the AppSpace page and use the **Create** button to create a AppSpace a2. You cannot view AppSpace a2 as you do not have access permissions for a2. The administrator has to grant permissions to access this AppSpace to enable custom users to see it. This is applicable to all entity types.
- While taking a backup of a domain, all entities within this domain are backed up irrespective of the permissions granted.

- The AppNodes, AppSpaces, and Application Archives count can be seen on the **Domain Management Page** irrespective of permissions granted to the user.

Administration Tasks and Reference

Administration tasks involve managing domains, AppSpaces, AppNodes, and applications.

The topics in this section show how to do administrative tasks from the BWAdmin command line and the Admin UI:

- To complete tasks from the BWAdmin command line, navigate to *BW_HOME*`\bin`. Type `bwadmin help` for a list of commands. For information, see bwadmin.
- To complete tasks from the Admin UI, register the TEA agent to the BWAgent and open the TEA URL. For more information, see Using the Admin UI.

Managing Domains

A domain is a logical group that provides an isolated environment for applications and their resources to reside. It provides an administrative boundary for an integration project. Each domain may share machines with other domains, but does not communicate with other domains. Domains include servers that may or may not be distributed over different machines and operating systems.

Creating a Domain

A domain comprises AppSpaces and AppNodes. Create a domain first, and then add the AppSpaces and AppNodes to the domain. The domain name is applied to all contained entities.

The following characters are allowed in the domain name:

- `A-Z`
- `a-z`
- `0-9`
- `-` (hyphen)
- `_` (underscore)

Illegal characters are stripped from the name.

The maximum length of a runtime entity name is 100 characters. If the maximum length is exceeded, the entity name is shortened to 100 characters.

BWAdmin Command Line

If the `-home` option is not specified in the `create` command (to set the path to the folder where all files related to the domain are managed), the domain is created in the default location under the *BW_HOME*`\domains` directory.

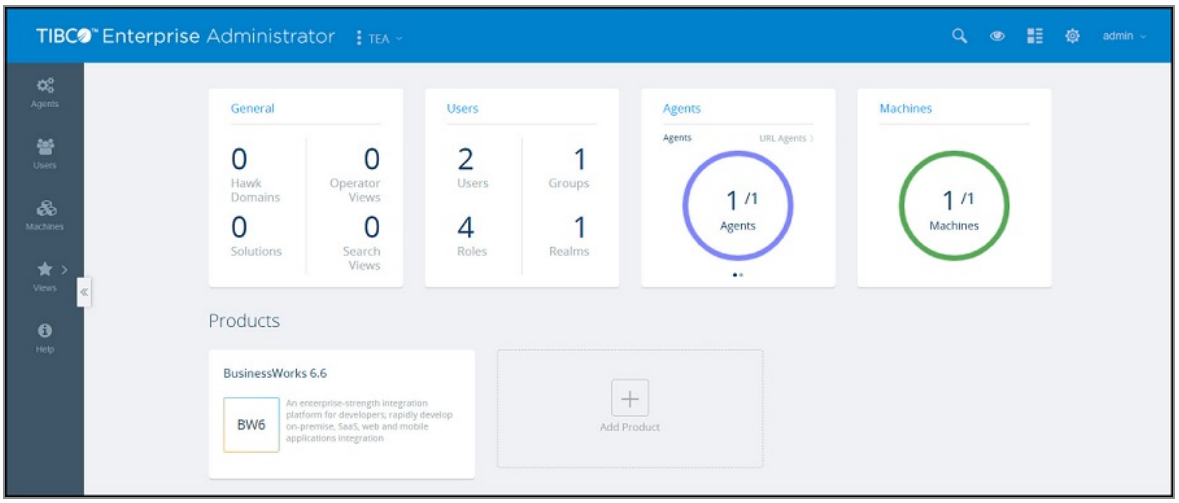
Run the following command to create a domain named MyDomain:

```
BW_HOME\bin>bwadmin create domain MyDomain
```

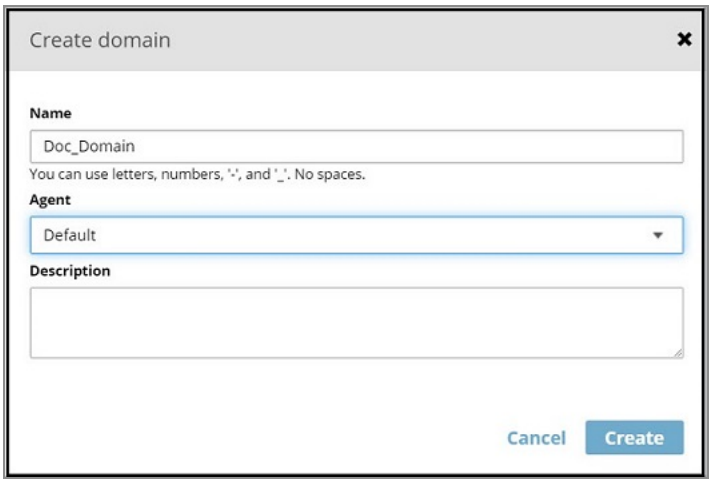
Admin UI

Procedure

1. Click the BusinessWorks product icon on the Admin UI home page.

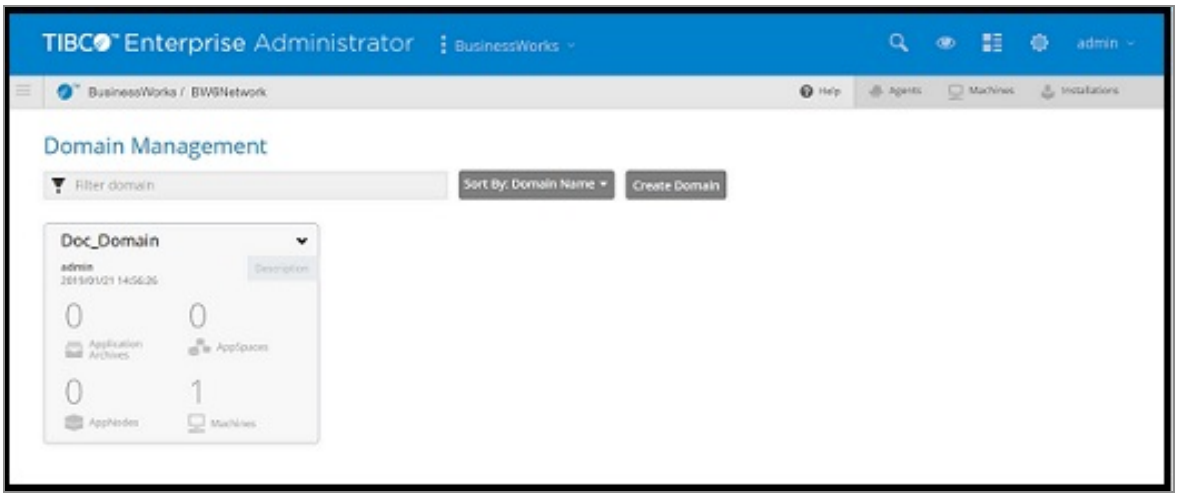


2. Click **Create domain**.
3. In the **Create domain** dialog, enter the domain name in the **Name** field.
4. Choose the agent registered with the TEA server from the **Agent** dropdown.



5. Click **Create**.

   The domain is displayed on the **Domain Management** page.



Deleting a Domain

Force delete a domain to remove all domain entities, including AppSpaces and AppNodes.

Note: A domain deletion cannot be reversed. After a domain force delete, the domain and all entities inside the domain are deleted.

BWAdmin Command Line

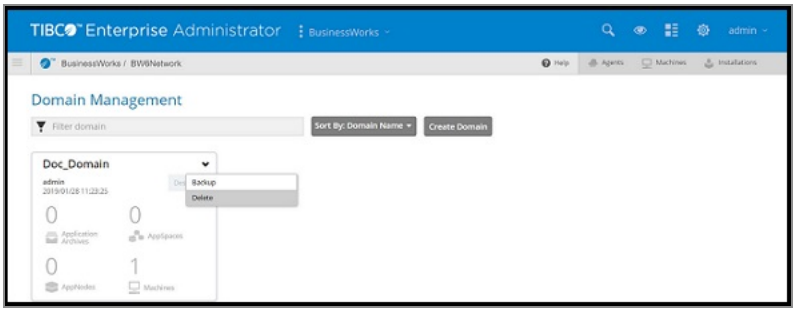
You can delete an empty domain or one that contains one or more AppSpaces.

Option	Command
To delete an empty domain	`BW_HOME\bin>bwadmin delete domain MyDomain`
To delete a domain that contains one or more AppSpaces	`BW_HOME\bin>bwadmin delete -force domain MyDomain`
To delete a domain using the timeout and force argument	`BW_HOME\bin>bwadmin delete -timeout xx(time in minutes) -force domain MyDomain` **Note:** The `-timeout` argument is valid only when the AppSpace is running. For more information, see Force Shutting Down an AppNode.

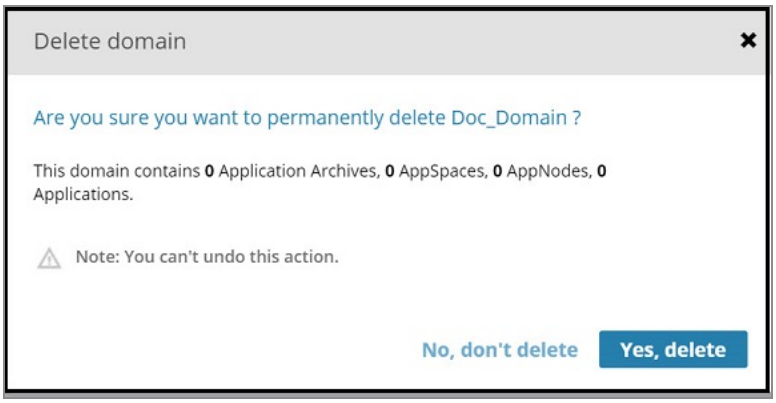
Admin UI

Procedure

1. Click the down arrow for the domain on the **Domain Management** page and choose **Delete**.



2. Click **Yes, delete** in the **Delete domain** dialog.



Backing Up and Restoring a Domain

Backing up a domain exports the current state of the specified domain and contained runtime entities to a BWAdmin command file. The entire domain is backed up, including remote BWAgents, if applicable to the specified domain. The command file can be provided to BWAdmin to recreate the domain. Output can be compressed to a ZIP file with the `-zipped` option.

BWAdmin Command Line

Procedure

1. To back up the current state of a domain, including profiles and archives, enter the `backup` command at the command line, using the `-s` option to identify the name of the destination file. Use the `domain` argument in the command line, with the name of the domain to back up. The domain can be either a local domain or a domain in a BWAgent in the agent network. By default, destination files are written to the current working directory.

   This example backs up domain `Machine2Domain` in a networked BWAgent to a command file named `machine2_domain.cmd`.

> **Note:** Use the `-noarchives` option to exclude archives uploaded to the domain from the backup. The references to the archives are included in the destination file. If needed, the paths in the destination file can manually be added to include archives in the restore.

The syntax is as follows where `-na` invokes the no archive option, and `-z` creates a zip file.

```
backup -na -z -s C:/Backup/archives.zip domain Domain_Name
```

```
backup -na -s C:/Backup/archives.cmd domain Domain_Name
```

```
BW_HOME\bin>bwadmin backup -s machine2_domain.cmd domain
Machine2Domain
```

If you are restoring to a different location, you need to update the command file as follows:

- The agent name points to `localhost` by default. You need to change this to the name of the machine that you are restoring to.
- Update the domain home to point to the absolute path to the new location.
- Update the path to the application archive (EAR) file to an absolute path.

2. To restore the domain,
   a. Enter the `BWAdmin` command, providing the name of the backup command file. The following example recreates the domain `Machine2Domain` and the contained runtime entities.

      ```
      BW_HOME\bin>bwadmin -f machine2_domain.cmd
      ```

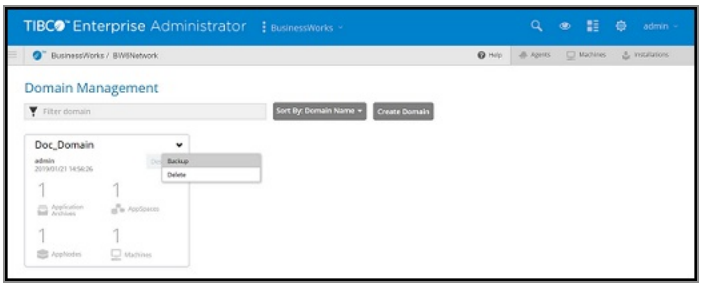
   b. Use the `bwadmin show domains` command from the command line to verify the restore.

Admin UI

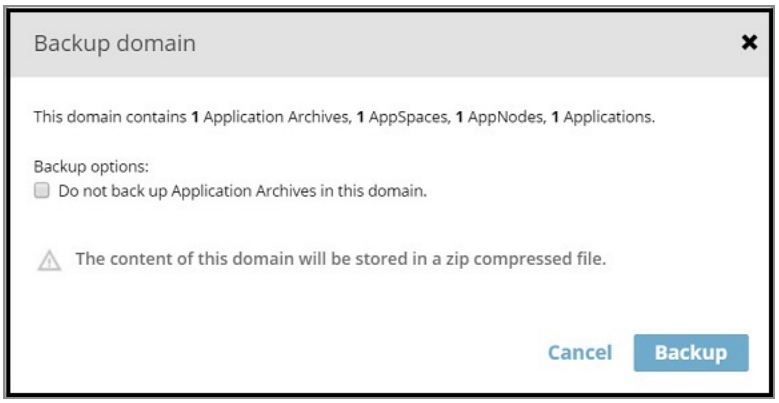
Procedure

1. Click the down arrow icon for the domain on the **Domain Management** page and

choose **Backup**.



2. Click **Backup** in the **Backup domain** dialog.
    a. To exclude archives from the backup, select the **Do not back up Application Archives in this domain** option.



The contents of the domain are written to a ZIP file that is downloaded to your computer. The filename is in the `Domain_backup_`*`domainName`*`.zip` format. The ZIP file contains a `.cmd` file that can be used to restore the environment.

Restoring the File System of a Domain

Restoring a domain restores the file system of the specified domain and all runtime entities in the domain to the state of the datastore.

Before you begin

- The name of the domain must be known in order to restore.
- The BWAgent must be running.

Procedure

1. To restore the file system for a domain and its runtime entities, enter the `restore` command at the command line, using the `domain` argument with the name of the domain to restore. The domain can be either a local domain or a domain in a BWAgent in the agent network. This example restores domain Machine2Domain in a networked BWAgent named Machine2.

   ```
   BW_HOME\bin>bwadmin restore -agent Machine2 domain Machine2Domain
   ```

2. To verify the restore, check the file system. Open the *BW_HOME*`\domains` folder. Look for a domain folder that matches the name of the domain.

Managing AppSpaces

An application is deployed to an AppSpace.

An AppSpace is a virtual pool of AppNodes where an application is deployed. When an application is deployed, the AppSpace starts the application on each of its AppNodes. More AppNodes can be added dynamically to the AppSpace to manage the load-balancing and fault tolerance needs for an application.

One or more applications can be deployed to an AppSpace.

A minimum number of AppNodes can be specified as a threshold for determining the AppSpace state. If the threshold falls below the minimum, the runtime state becomes `Degraded`.

When an application deployed to an AppSpace runs, and scalability is enabled, all the AppNodes in the AppSpace are started and share the load for the application. If scalability is turned off for a deployed application, the application runs on just one AppNode. For more information, see Fault Tolerance.

Creating an AppSpace

An AppSpace is created under a domain, which must exist before adding an AppSpace to it. An AppSpace contains one or more AppNodes. The domain name applies to the AppSpace.

The following characters are allowed in the AppSpace name:

- `A-Z`

- a-z
- 0-9
- – (hyphen)
- _ (underscore)

Illegal characters are stripped from the name.

The maximum length of a runtime entity name is 100 characters. If the maximum length is exceeded, the entity name is shortened to 100 characters.

The appspace_config.ini_template file is available as a template to create a config.ini file for an AppSpace. If you want to create an AppSpace with some different configuration than the standard configuration, update the appspace_config.ini_template file.

The appspace_config.ini_template is stored at {BW_HOME}\<version>\config location.

BWAdmin Command Line

To create an AppSpace named MyAppSpace in the domain MyDomain, run the following command at the command line:

```
BW_HOME\bin>bwadmin create -d MyDomain -minNodes 1 appspace MyAppSpace
```

The MyAppSpace AppSpace is created in the domain MyDomain that exists on the machine where the BWAgent is running. Use the -agent option to create an AppSpace running on a remote machine. Run the show agents command on the remote machine to get the agent name. If you are in local mode, the agent is not supported.

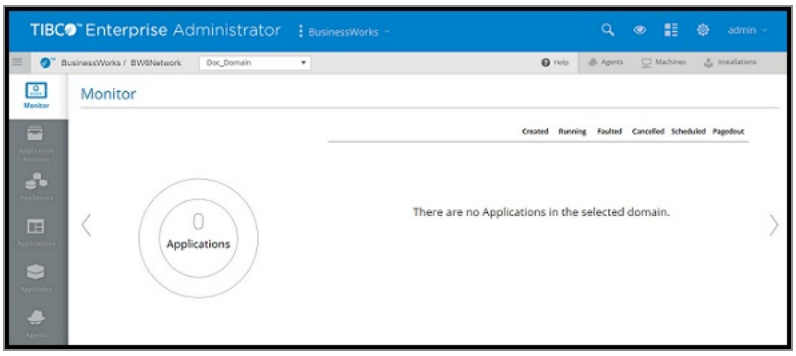
Admin UI

To create an AppSpace using the Admin UI:

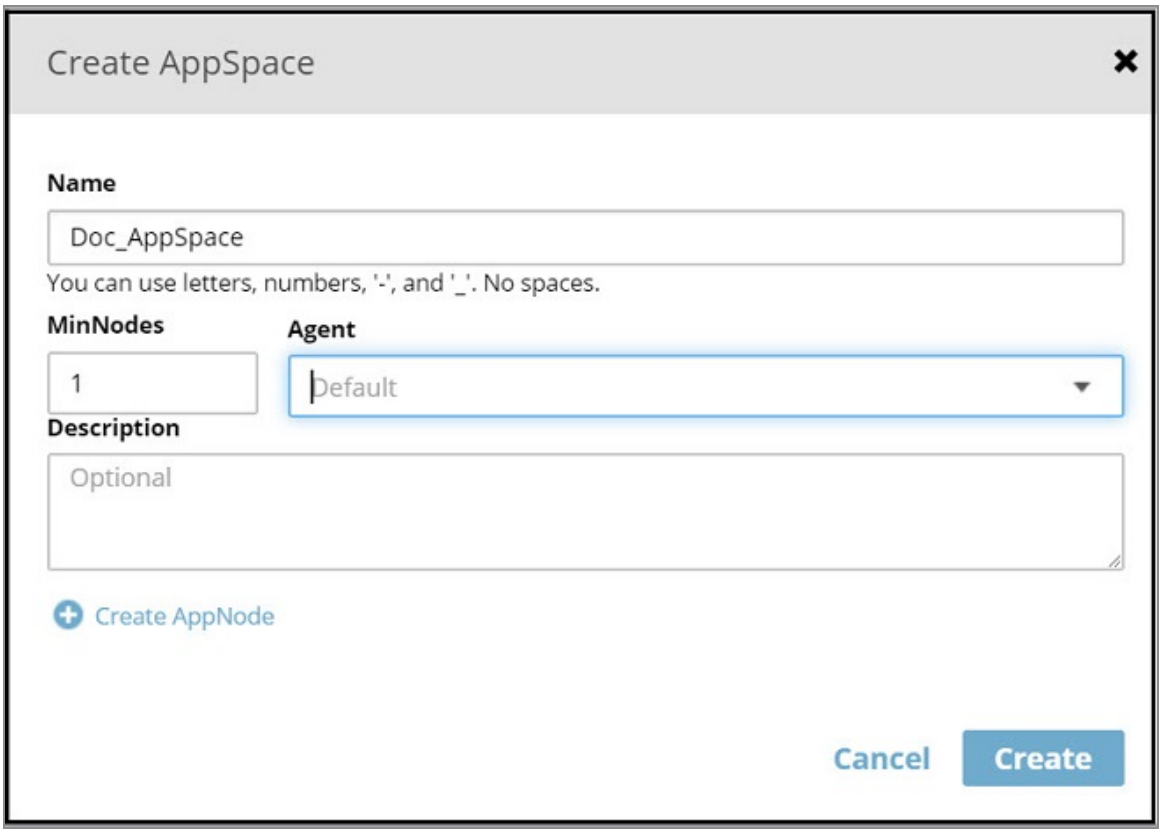
Procedure

1. On the **Domain Management** page, click the domain you want to add the AppSpace to.

   The **Monitor** page is displayed, showing that no runtime entities exist in the domain.



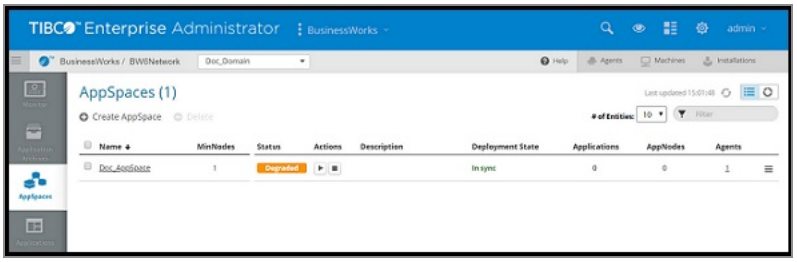
2. Click **AppSpaces** on the side bar to open the **AppSpaces** page.
3. Click **Create AppSpace** to open the **Create AppSpace** dialog. Enter the following information.
    - **Name**: AppSpace name.
    - **MinNodes**: Minimum number of AppNodes for this AppSpace. Default is 1. The AppSpace status is set to **Degraded** if the minimum number of AppNodes is not created.
    - **Agent**: The BWAgent registered with the TEA server.
    - **Description**: Optional description.



Click **Create AppNodes** in the **Create AppSpace** dialog to create AppNodes. You can also create AppNodes from the **AppNodes** Admin UI page. For information, see Creating an AppNode.

4. Click **Create.**

   The AppSpace is created and displayed on the **AppSpaces** page. The AppSpace status is set to Degraded as there are no AppNodes yet to satisfy the minimum requirement.



Starting an AppSpace

To run applications in an AppSpace, start the AppSpace.

Note: If an AppSpace does not contain any AppNodes, it does not start. The AppSpace status is set to `Degraded`. The minimum number of nodes must be created in order for the AppSpace to start.

BWAdmin Command Line

To start the AppSpace MyAppSpace in the domain MyDomain, run the following command at the command line:

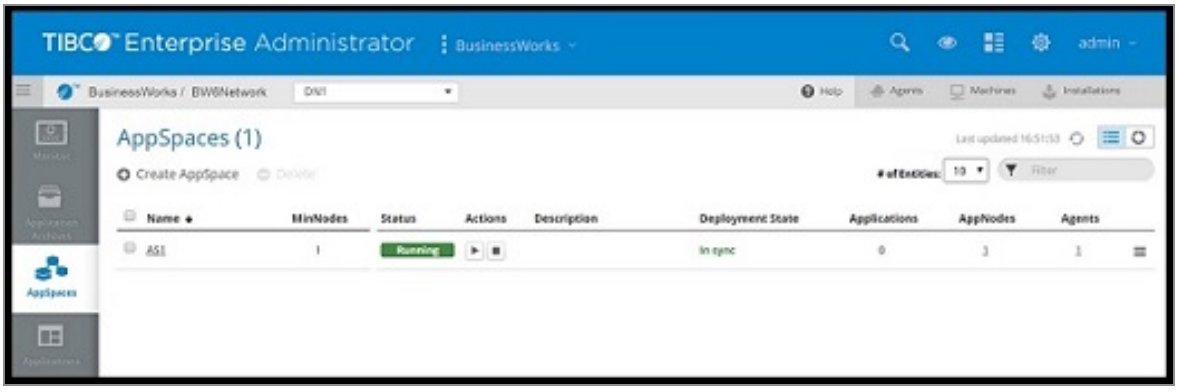
```
BW_HOME\bin>bwadmin start -d MyDomain appspace MyAppSpace
```

Admin UI

Procedure

1. On the **AppSpaces** page for the domain, click the **Start** icon for the AppSpace you want to start.

   If a minimum number of nodes exists, the status displays as **Starting**, a transient state, then **Running**.



Editing an AppSpace Configuration

You can edit the configuration for a running AppSpace from the Admin UI.

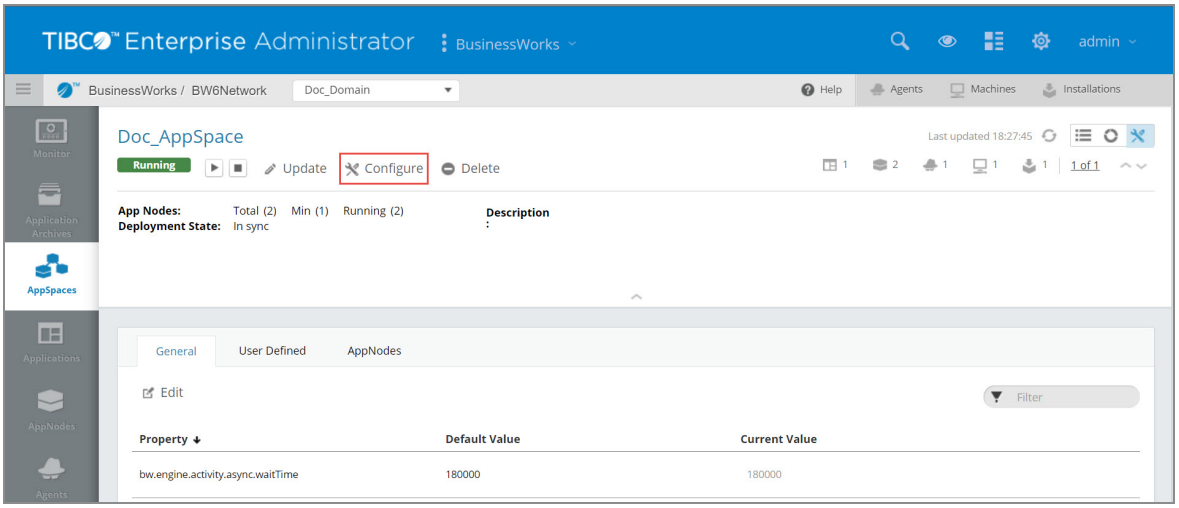
Admin UI

For information about some of the properties, you can configure for an AppSpace using the Admin UI, see:

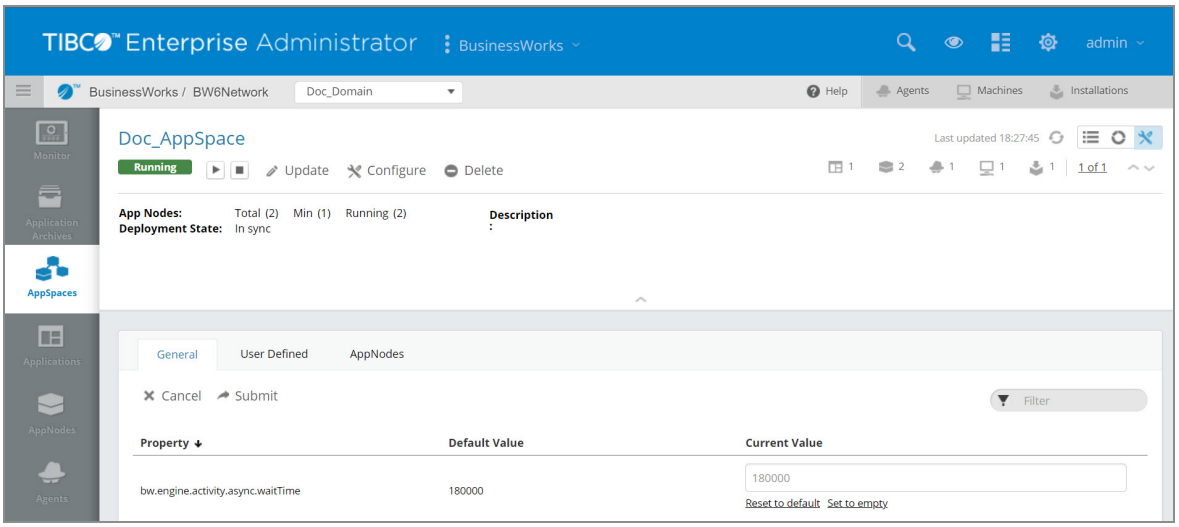
- Statistics Collection
- Engine Persistence Modes
- Tuning

Procedure

1. Select the AppSpace you want to configure on the **AppSpaces** page.
2. Click **Configure**. The **AppSpace Properties** page is displayed. Use the **General** tab to edit AppSpace properties.



3. Click **Edit** to open the tab for editing, and click **Submit** when you are done.



Note: You can also edit user-defined properties on the **User Defined** tab or AppNode properties on the **AppNode** tab.

Viewing AppSpace States

An AppSpace has two states: Deployment and Runtime.

The Deployment state can have the following statuses:

AppSpace Deployment Statuses

Status	Description
In-Sync	The AppSpace is synchronized with its BWAgents.
Out-of-Sync	The AppSpace is out of synchronization. The out-of-sync state may occur when: • a BWAgent is not reachable due to network failure, or • the BWAgent configuration may not have been applied remotely.

The Runtime state can have the following statuses:

AppSpace Runtime Statuses

Status	Operations Allowed in This Status	Description
Running	Stop	The minimum threshold of AppNodes configured for this AppSpace are running.
Stopped	Start, Delete	None of the AppNodes configured for this AppSpace are running.
Degraded	Stop	The number of AppNodes for this running AppSpace falls below the minimum specified threshold. **Note:** The state also occurs when the AppSpace is not running.

Note: AppSpaces do not have a starting state. However, AppNodes have their own lifecycle and may go from `Starting` to `Stopped`.

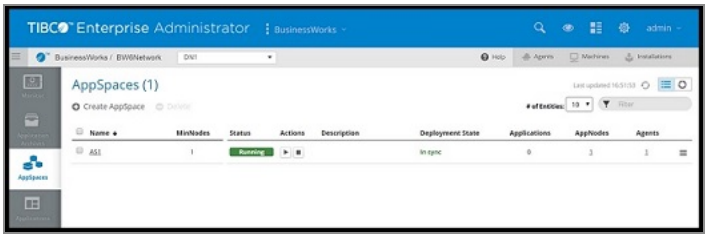
BWAdmin Command Line

To view the status of the AppSpace `MyAppSpace` in the domain `MyDomain`, run the following command at the command line:

```
BW_HOME\bin>bwadmin show -domain MyDomain appspace MyAppSpace
```

Admin UI

Navigate to the **AppSpace** page and view the **Status** column.



Stopping an AppSpace

When an AppSpace is stopped, all applications and AppNodes running in the AppSpace stop.

BWAdmin Command Line

To stop the AppSpace `MyAppSpace` in the domain `MyDomain`, run the following command at the command line:

```
BW_HOME\bin>bwadmin stop -d MyDomain appspace MyAppSpace
```

To force shut down the AppSpace `MyAppSpace` in the domain `MyDomain`, run the following command at the command line:

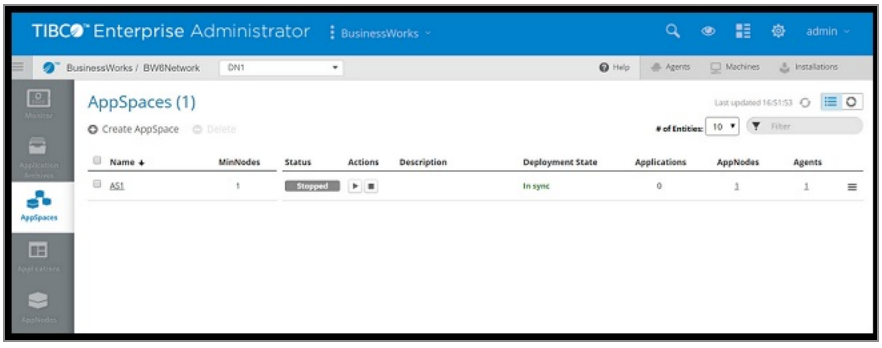
```
BW_HOME\bin>bwadmin stop -timeout xx(time in minutes) -domain MyDomain
appspace MyAppSpace
```

For more information, see Force Shutting Down an AppNode.

Admin UI

On the **AppSpaces** page, click the **Stop** icon for the AppSpace you want to stop.

The status for the AppSpace changes from **Running** to **Stopping**, a transient state, then Stopped.



Deleting an AppSpace

An AppSpace can be deleted if it does not have associated AppNodes. If it contains AppNodes, you can force delete it.

BWAdmin Command Line

To delete the AppSpace `MyAppSpace` in the domain `MyDomain`, run the following command at the command line:

```
BW_HOME\bin>bwadmin delete -d MyDomain appspace MyAppSpace
```

If the AppSpace has an attached AppNode, the `delete appspace` command fails. You can delete the attached AppNode and retry the `delete appspace` command or use the `delete appspace` command with the `-force` option.

```
BW_HOME\bin>bwadmin delete -force -domain MyDomain appspace MyAppSpace
```

To delete the AppSpace `MyAppSpace` in the domain `MyDomain` and shut down the running AppNodes forcefully, run the following command at the command line:

```
BW_HOME\bin>bwadmin delete -timeout xx(time in minutes) -force -domain
MyDomain appspace MyAppSpace
```

For more information, see Force Shutting Down an AppNode.

To delete all AppSpaces in the domain `MyDomain`, run the following command at the command line:

```
BW_HOME\bin>bwadmin delete -d MyDomain -all appspace
```

If any of the AppSpace in the domain `MyDomain` contains AppNode,

- Either first delete those AppNodes and then run

  ```
  BW_HOME\bin>bwadmin delete -d MyDomain -all appspace
  ```

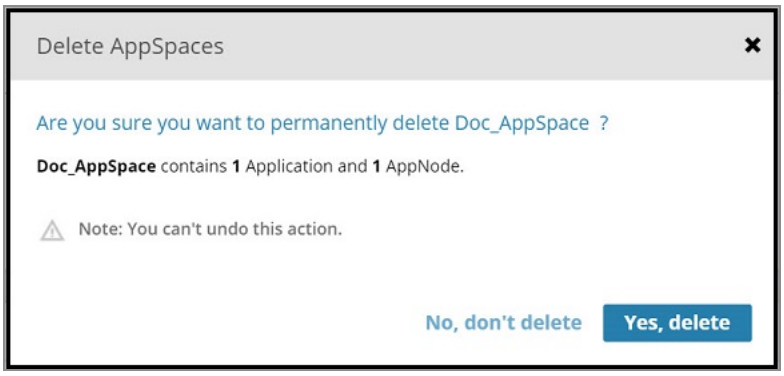
- Or, if you want to delete all AppSpaces including AppNodes forcefully, run the following command:

```
BW_HOME\bin>bwadmin delete -d MyDomain -all -force appspace
```

Admin UI

Procedure

1. On the **AppSpaces** page, select the checkmark next to the AppSpace you want to delete.
2. Click **Delete**.
3. Click **Yes, delete** in the **Delete AppSpaces** dialog. The dialog message displays the number of applications and AppNodes that are deleted.



Backing Up and Restoring an AppSpace

Backing up an AppSpace exports the current state of the specified AppSpace to a BWAdmin command file. The command file can be provided to BWAdmin to recreate the AppSpace. Output can be compressed to a ZIP file with the `-zipped` option.

Procedure

1. To back up the current state of an AppSpace, enter the `backup` command at the command line, using the `-s` option to identify the name of the destination file. Use the `-domain` option with the `appspace` argument in the command line, with the name of the AppSpace to back up. The AppSpace can be either a local AppSpace or an AppSpace in a BWAgent in the agent network. By default, destination files are written to the current working directory.

This example backs up AppSpace `MyAppSpace` in Domain `MyDomain` to a command file named `myappspace.cmd`

```
BW_HOME\bin>bwadmin backup -s myappspace.cmd -domain MyDomain
appspace MyAppSpace
```

2. To restore the AppSpace:
   a. Enter the `bwadmin` command at the command line, providing the name of the backup command file. The following example recreates the AppSpace `MyAppSpace`.

      ```
      BW_HOME\bin>bwadmin -f myappspace.cmd
      ```

      If you are restoring to a different location, you need to update the command file as follows:

      - The agent name points to `localhost` by default. You need to change this to the name of the machine that you are restoring to.
      - Update the domain home to point to the absolute path to the new location.
      - Update the path to the application archive (EAR) file to an absolute path.

   b. Use the `bwadmin show appspaces` command from the command line, with the `-domain` option to verify the restore.

Restoring the File System of an AppSpace

Restoring an AppSpace restores the file system of the specified AppSpace and all runtime entities in the AppSpace to the state of the datastore.

Before you begin

- The name of the containing domain and the name of the AppSpace must be known in order to restore.
- The BWAgent must be running.

Procedure

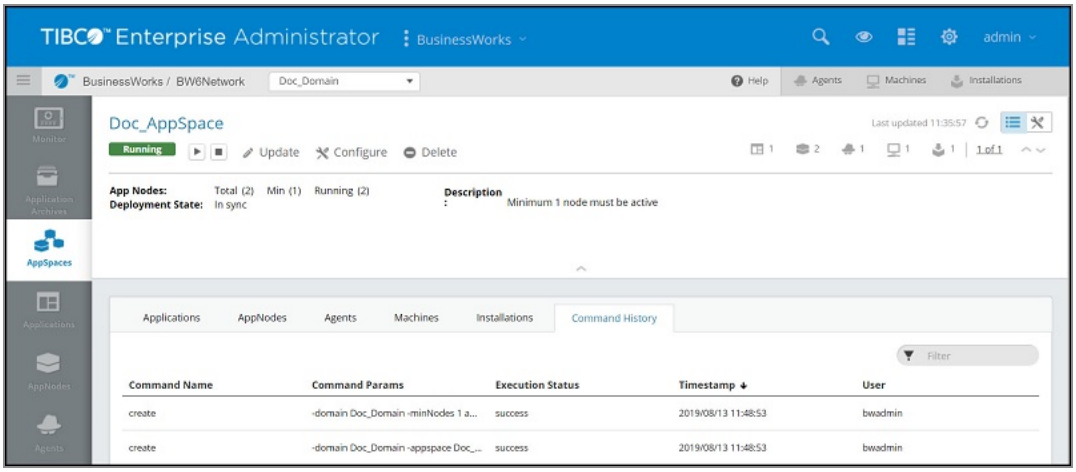
1. To restore the file system for an AppSpace and the runtime entities in the AppSpace, open a terminal and navigate to *BW_HOME*\bin.
2. Enter the restore command from the command line, using the -domain option with the appspace argument specifying the name of the AppSpace to restore. This example restores AppSpace MyAppSpace in domain MyDomain.

```
BW_HOME\bin>bwadmin restore -d MyDomain appspace MyAppSpace
```

3. To verify the restore, check the file system. Open the *BW_HOME*\domains folder. Check for the named AppSpace folder under: *BW_HOME*\domains*domain_name*\appspaces

Command History

Open the **Command History** tab to view the commands or operations that were performed on an AppSpace.



Managing AppNodes

An AppNode is a runtime entity for hosting application modules and libraries.

An AppNode represents a physical engine process that is launched when an application starts to run.

- Install ActiveMatrix BusinessWorks on each machine hosting an AppNode.

- One or more AppNodes can be created in an AppSpace.

Creating an AppNode

An AppNode is created under an AppSpace. The domain and AppSpace name apply to the AppNode.

Multiple AppNodes can be created for an AppSpace.

When creating an AppNode that is on a remote machine, ensure that:

- The remote BWAgent is a part of the network.
- The name of the BWAgent running on the remote machine is specified.

Note: When an AppNode is created, do not specify the OSGi port. Only open this port for debugging when enabling the OSGi console on an AppNode. For more details, see Enabling the OSGi Console for an AppNode.

The following characters are allowed in the AppNode name:

- `A-Z`
- `a-z`
- `0-9`
- – (hyphen)
- _ (underscore)

Illegal characters are stripped from the name.

The maximum length of a runtime entity name is 100 characters. If the maximum length is exceeded, the entity name is shortened to 100 characters.

The `appnode_config.ini_template` file is available as a template to create a `config.ini` file for an AppNode. If you want to create an AppNode with some different configuration than the standard configuration, update the `appnode_config.ini_template` file.

The `appnode_config.ini_template` is stored at the `{BW_HOME}\<version>\config` location.

BWAdmin Command Line

Use the `create` command to create an AppNode.

The BWAgent must be running. Issue the following command to create an AppNode named `MyAppNode` in domain `MyDomain` and AppSpace `MyAppSpace`:

```
BW_HOME\bin>bwadmin create -d MyDomain -a MyAppSpace -httpPort 2222
appnode MyAppNode
```

The `httpPort` option is required for an AppNode. If the specified port is already in use, an error occurs and the AppNode is not created. To get a list of defined AppNodes for a given domain, with port numbers, with the `show` command: `show -d <DomainName> appnodes`

The following command creates an AppNode `MyAppNodeOnMac` in the domain `MyDomain` and AppSpace `MyAppSpace` on a remote machine whose agent name is `Machine2`.

```
BW_HOME\bin>bwadmin create -d MyDomain -a MyAppSpace -httpPort 2222 -
agent Machine2 appnode MyAppNodeOnMac
```

Note:

- To create an AppNode on a remote machine, the member name of the BWAgent on the machine where the AppNode is run must be known in advance. Get the member name value by invoking the BWAdmin `show agents` command on the remote machine.
- The validation of the HTTP ports is available by running the command `validateport [options] port`. For example,

  ```
  bwadmin[admin]> validateport 2233
  TIBCO-BW-ADMIN-CLI-300342: HttpPort [2233] is available
  within BW scope
  ```

  Or

  ```
  bwadmin[admin]> validateport 344566
  TIBCO-BW-ADMIN-CLI-500338: HttpPort is not valid
  ```

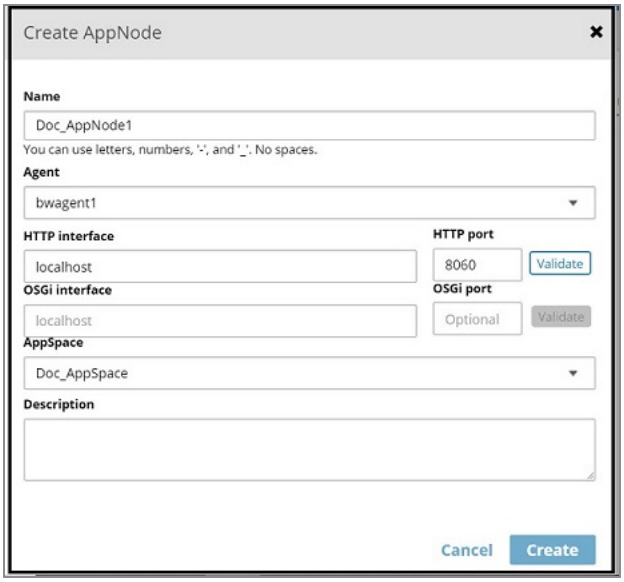
  To know more about the `validateport` command, run the command `validateport --help`.

Admin UI

Procedure

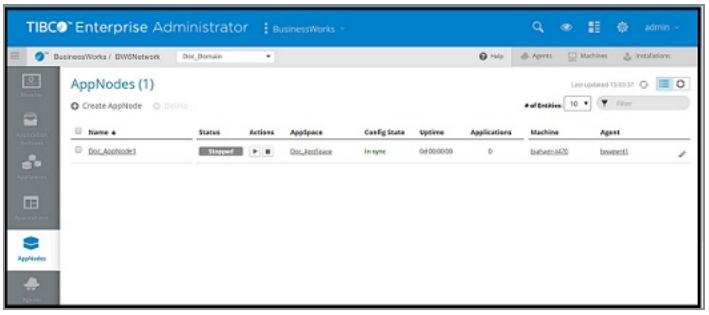
1. Click **AppNodes** to open the **AppNodes** page.
2. Click **Create AppNode** to open the **Create AppNode** dialog. Enter the following information:
   - **Name**: AppNode name.
   - **Agent**: The BWAgent registered with the TEA server.
   - **HTTP interface**: The HTTP interface for the AppNode.
   - **HTTP port**: The HTTP port for the AppNode. Click **Validate** to see if the port is available.

     Note: The **Validate** button validates the HTTP ports within the ActiveMatrix BusinessWorks scope only.
   - **OSGi interface**: The OSGi interface for the AppNode. Open this port only for debugging sessions.
   - **OSGi port**: The OSGi port for the AppNode. Open this port only for debugging sessions.
   - **AppSpace**: The AppSpace for this AppNode.
   - **Description**: Optional description.



3. Click **Create**.

   The AppNode is created and displayed on the **AppNodes** page. The AppNode status is set to Stopped.



Creating an AppNode using a Secured Port (HTTPS)

Use the `create` command to create an HTTPS AppNode.

The BWAgent must be running with the HTTPS port. To create an AppNode named `MyAppNode` in domain `MyDomain` and AppSpace `MyAppSpace`, run the following command:

```
BW_HOME\bin>bwadmin create -d MyDomain -a MyAppSpace -httpsPort 7041 -
ksPath <keystore path> -ksPwd <keystore password> -sslPwd <ssl password>
appnode MyAppNode
```

The HTTPS port, keystore path, keystore password, and SSL password fields are required for an HTTPS AppNode. If the specified port is already in use, an error is displayed and the HTTPS AppNode cannot be created. To get a list of defined HTTPS AppNodes for a given domain with port numbers, use the following `show` command:

```
show -d <DomainName> appnodes
```

To start the BWAgent with HTTPS port, perform the following steps:

Procedure

1. In the `bwagent.ini` file, add the following property and assign a suitable port for the HTTPS communication:

   ```
   bw.appnode.agent.https.communication.port=56565
   ```

2. In the `bwagent.ini` file, modify the following keystore properties:

```
bw.agent.https.keystorepassword=<Keystore password>
```

```
bw.agent.https.keystorepath=<keystore_path>
```

> **Note:** While creating an HTTPS AppNode, use the same keystore that you have used to start the BWAgent via the HTTPS port.

BWAdmin Command Line

To create an HTTPS AppNode with the BWAdmin Command Line, run the following command:

```
BW_HOME\bin>bwadmin create -d MyDomain -a MyAppSpace -httpsPort 7041 -
ksPath <keystore path> -ksPwd <keystore password> -sslPwd <ssl password>
appnode MyAppNode
```

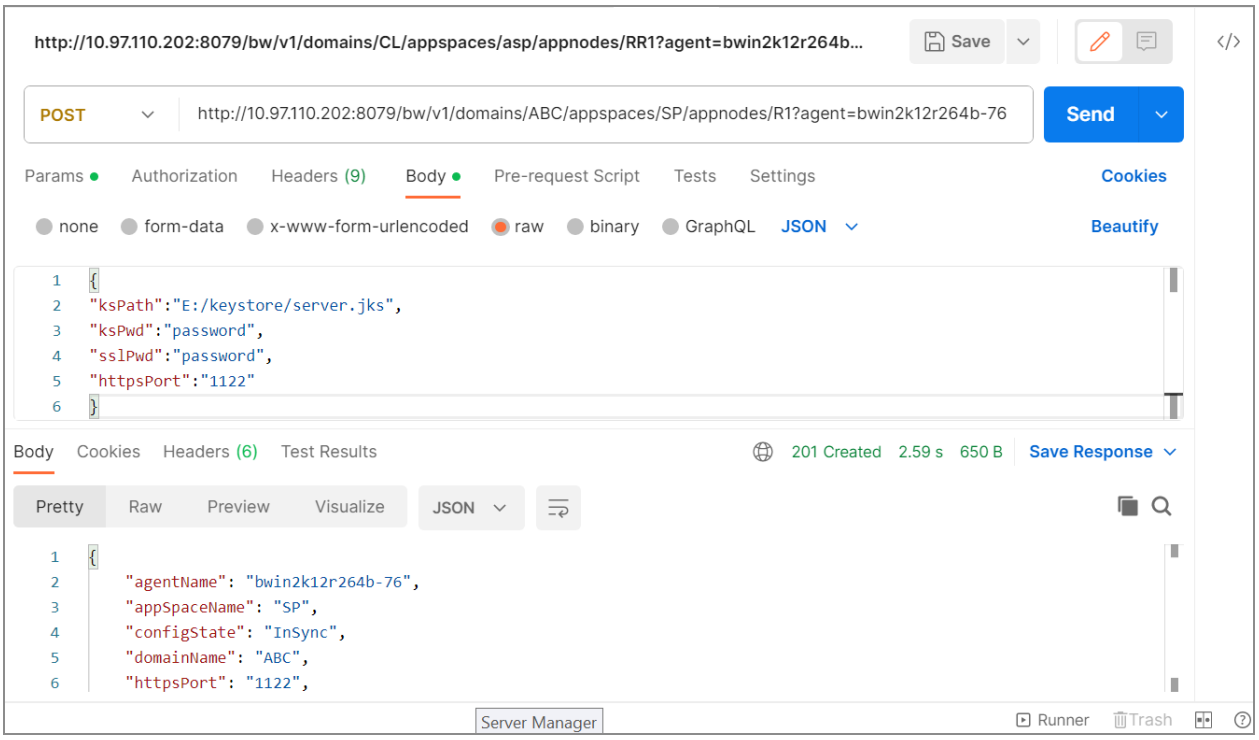
where,

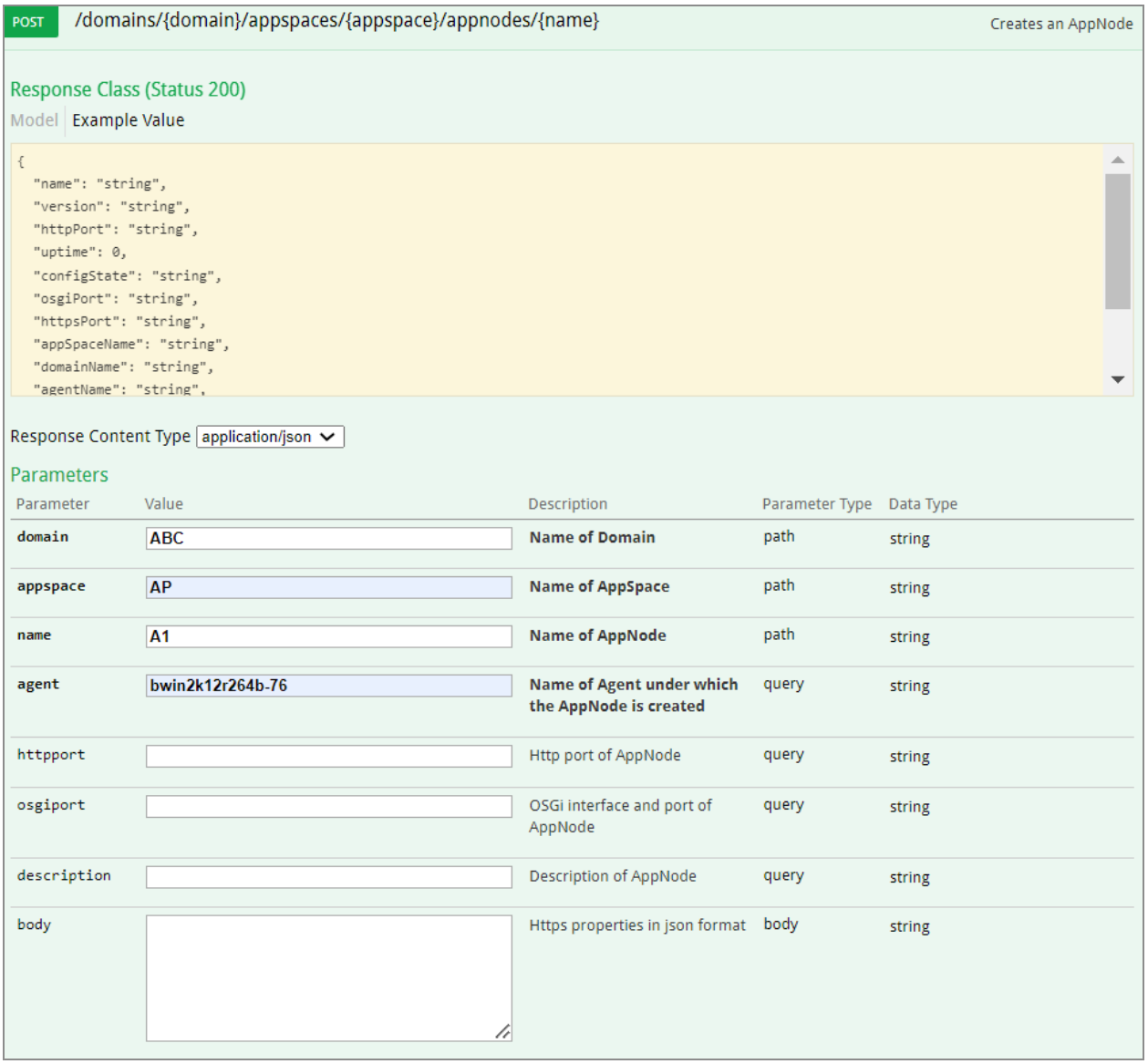
- `-httpsPort`: HTTPS port
- `-ksPath`: Keystore path
- `-ksPwd`: Keystore password
- `-sslPwd`: SSL password

REST API

To perform the same operation via the REST API, follow the below API to create an HTTPS AppNode:

- `http://<host>:8079/bw/v1/domains/<DomainName>/appspaces/<AppSpaceName>/appnodes/<AppNodeName>?agent=<AgentName>`


http://10.97.110.202:8079/bw/v1/domains/CL/appspaces/asp/appnodes/RR1?agent=bwin2k12r264b...
Save
POST
http://10.97.110.202:8079/bw/v1/domains/ABC/appspaces/SP/appnodes/R1?agent=bwin2k12r264b-76
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"ksPath":"E:/keystore/server.jks",
"ksPwd":"password",
"sslPwd":"password",
"httpsPort":"1122"
}
Body
Cookies
Headers (6)
Test Results
201 Created
2.59 s
650 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"agentName": "bwin2k12r264b-76",
"appSpaceName": "SP",
"configState": "InSync",
"domainName": "ABC",
"httpsPort": "1122",
Server Manager
Runner
Trash



AdminUI

Currently, the creation of the HTTPS AppNode via AdminUI is not supported.

Limitations

- When starting the BWAgent securely, you can create an AppNode only with a secured port. If you configure the AppNode to an HTTP (unsecured) port, it fails to work.
- When starting the BWAgent in an unsecured way, you can create an AppNode only

with an unsecured port. If you configure the AppNode to an HTTPS (secured) port, it fails to work.

- When starting the BWAgent and AppNode with a secured connection, while restarting only the BWAgent with an unsecured port, the AppNode fails to communicate with the BWAgent.
- When two BWAgents are in the network (one is secured and the other is unsecured), you can create secured AppNodes with the secured BWAgent and unsecured AppNodes with the unsecured BWAgent. In other words, secured and unsecured AppNodes are in the network.
- Consider creating an AppNode on an unsecured BWAgent from a secured BWAgent and both of them are in the network. While providing the keystore details, if you provide details of the secured BWAgent machine, the AppNode on the unsecured BWAgent fails to start. In this case, you must provide keystore details from the unsecured BWAgent machine.
- Consider upgrading from ActiveMatrix BusinessWorks 6.6.0 to 6.10.0. When taking a backup of an unsecured environment (6.6.0) and after restoring on a secured environment (6.10.0), it fails.

  Workaround:

  1. Take the backup of ActiveMatrix BusinessWorks 6.6.0.
  2. Start version 6.10.0 in an unsecured way.
  3. Perform the restore of version 6.6.0 on version 6.10.0.
  4. Secure the 6.10.0 environment and restart the BWAgent.

  Also, a utility is required that updates all the AppNodes in a secured way when you restart the BWAgent.
- Currently there is a provision to provide a keystore path while creating a secured AppNode. For creating a secured AppNode on a remote machine, you must have a keystore file on your local machine.
- When configuring the HTTPS port from the BWAdmin CLI, you must provide all three parameters (keystore path, keystore password, and SSL password), else it fails to support.
- When creating an AppNode with the `keystore1.jks` file and starting the BWAgent with HTTPS configurations along with another keystore `keystore2.jks`, the AppNode fails to start.

Important:
- You cannot give an obfuscated password through the CLI while creating a secured AppNode.
- Update the AppSpace level configuration when creating a secured AppNode.

Starting an AppNode

Use the `start` command to start an AppNode manually.

When an AppSpace is started, all AppNodes associated with the AppSpace automatically start.

Note: By default, the value for the `bw.engine.shutdownOnFailure` property is true in the AppSpace `config.ini` file. This ensures that the AppNode does not start when there are any issues when starting the bwengine. You can also configure the property at the AppNode, or the AppSpace level.

BWAdmin Command Line

Run the following command at the command line to start the `MyAppNode` AppNode:

```
BW_HOME\bin>bwadmin start -d MyDomain -a MyAppSpace appnode MyAppNode
```

Tip: If the AppNode is not gracefully shut down, it could corrupt the /config folder.

Configure the `bw.appnode.clean.config.folder.on.startup` property in the AppNode, or the AppSpace `config.ini` file.

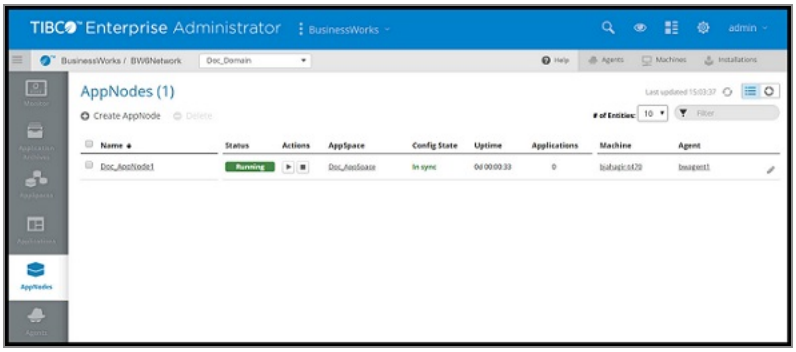
To create a new /config folder every time the AppNode starts, set the `bw.appnode.clean.config.folder.on.startup` property to true. If you set the property to `false`, or it is undefined, the `/config` folder is not deleted when the AppNode starts.

Admin UI

Procedure

1. On the **AppNodes** page, click the **Start** icon for the AppNode.

   The status for an AppNode is displayed as **Starting**, a transient state, then **Running**.



Editing an AppNode Configuration

You can edit the configuration for a running AppNode from the Admin UI. Changes are applied when you restart the AppNode.

Admin UI

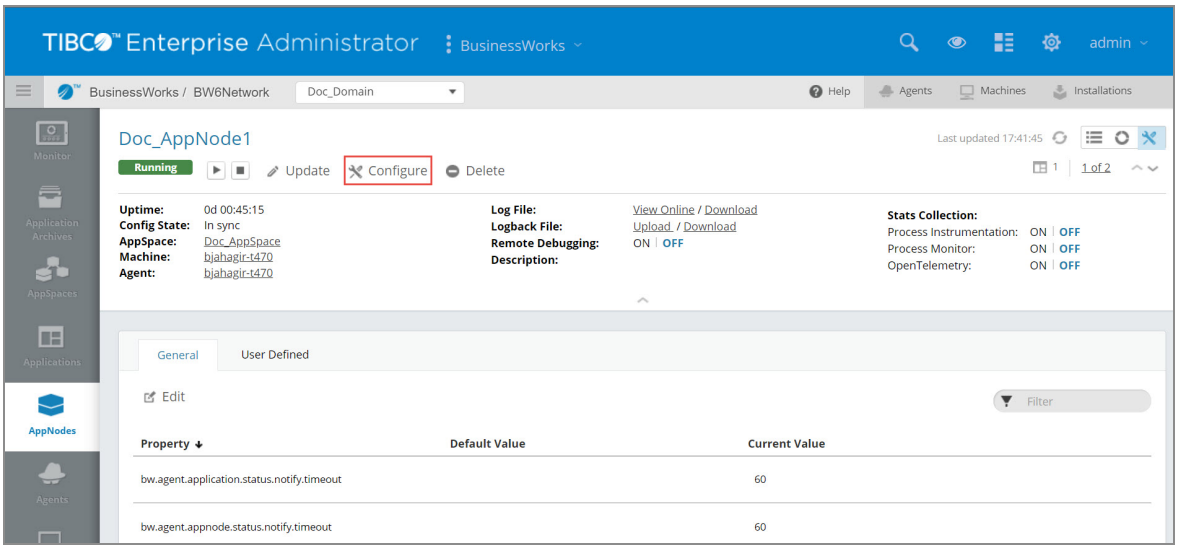
For information about some of the properties you can configure for an AppNode using the Admin UI, see:

- Application Statistics Collection
- Engine Persistence Modes
- Engine and Job Tuning
- Viewing Endpoints, Components, Processes, and Command History
- AppNode Logging

Procedure

1. Select the AppNode you want to configure on the **AppNodes** page.
2. Click **Configure**. The **AppNode Properties** page is displayed. Use the **General** tab to

edit AppNode properties.



3. Click **Edit** to open the tab for editing, and click **Submit** when you are done. You need to restart the AppNode to apply the changes. The AppNode status is set to `Out of sync` until the AppNode is restarted.

   You can also edit user-defined properties on the **User Defined** tab.

Auto Collecting Engine Data

The collection of data require multiple engine API (OSGi commands). These APIs are invoked internally and output is exported in file format at a specified location.

A REST API is provided to collect an engine or AppNode data. Invoke the REST API as POST: `http://<host>:<port>/bw/framework.json/collect/`.

The engine data collected for an AppNode for ActiveMatrix BusinessWorks is stored at *<user.dir>* `\..\debug\APPNODE_DATA_<TIME_STAMP>.zip` where, *<user.dir>* is of the form:

`$BW_HOME\bw\<version>\domains\<domain_name>\appnodes\<appspace_name>\<appnode_name>\bin`

BWAdmin Command Line

Run the following command at the command line to collect AppNode's data:

1. In a terminal, navigate to *BW_HOME*`\bin` and type bwadmin.

2. Go to **MyDomain**.

```
bwadmin[admin]> cd MyDomain
```

3. Go to **MyAppSpace**.

```
bwadmin[admin@MyDomain]> cd MyAppSpace
```

4. Start the AppNode, if it is not already running:

```
bwadmin[admin@MyDomain/MyAppSpace]> start appnode MyAppNode
```

5. Go to **MyAppNode**.

```
bwadmin[admin@MyDomain/MyAppSpace]> cd MyAppNode
```

6. Run the `collectappnodedata` command.

```
bwadmin[admin@MyDomain/MyAppSpace/MyAppNode]>collectappnodedata
[options] [operation]
```

The following options are available:

Option	Description
`-o, -override`	Delete all previously created data files. Generate new files as per the selected operation. It has two options, true or false. The default option is true.
`-i, -input`	Input list of operations to be performed. Comma-separated list without space. Sample input: "THREAD_DUMP", "HEAP_DUMP", "VM_ARGUMENTS", "ENVIRONMENT_VARIABLES", "THREAD_SNAPSHOT", "MEMORY_SNAPSHOT", "SYSTEM_PROCESS_INFORMATION", "SYSTEM_PROPERTIES", "CPU_INFORMATION", "osgiCommand1", "osgiCommand2"

Option	Description
`-d, -domain`	Domain name
`-p, -path`	Output directory path
`-n, -appnode`	Name of an AppNode
`-a, -appspace`	AppSpace name, Applicable when an entity is an AppNode
`-al, -all`	Download all files from the specified directory path.
`-dp, -downloadpath`	Download all files from the specified directory path. It has two options true or false. The default option is false.
`-dd, -downloadanddelete`	Delete file after download. It has two options, true or false. The default option is false.
`--help`	Display this help message

The following operations are available:

Operation	Description

`ALL`
Admin CLI command:
`collectappnodedata -o false -p "D:/appNode/data/" ALL`
	Available options:
	• `override`: [optional] Override previously created data. The default value is true.
	• `path`: [optional] Set the output directory path.
	Output:
	All default set of operations is run.
	Note: SYSTEM_PROCESS_INFORMATION is not run when running in TIBCO Business Studio for BusinessWorks.

Operation	Description

`INCLUDE`
 	
Admin CLI command:
 	
`collectappnodedata -p "D:/appNode/data/" -i "command1","command2" INCLUDE`
	Only the set of operations given as an input are run.
	
	Available options:
	
	• `override`: [optional] Override previously created data. The default value is true.
	• `path`: [optional] Set the output directory path.
	• `input`: Set of operations to run. Comma-separated list for admin CLI command and JSON list for REST API.
	
	Output:
	
	For example, if the input list is `la`, `lp`, "`lapi *`", `thread_dump`, then only these four operations are run.
	
	Note: Input list is mandatory to run this operation.

`EXCLUDE`
 	
Admin CLI command:
 	
`collectappnodedata -p "D:/appNode/data/" -i "command1","command2" EXCLUDE`
	All default set operations excluding the set of operations given as input are run.
	
	Available options:
	
	• `override`: [optional] Override previously created data. The default value is true.
	• `path`: [optional] Set the output directory path.
	• `input`: [optional] Set of operations to be run. Comma-separated list for admin CLI command and JSON list for REST API.
	
	Output:
	
	For example, if the input list is `la`, `lp`, "`lapi *`", `thread_dump`, then all default set operations without these four operations are run. The following operations are run:
	
	["HEAP_DUMP", "VM_ARGUMENTS", "ENVIRONMENT_VARIABLES", "SYSTEM_PROPERTIES", "THREAD_

Operation	Description
	SNAPSHOT", "MEMORY_SNAPSHOT", "SYSTEM_PROCESS_INFORMATION","CPU_INFORMATION", "LMETRICS", "LCFG", "LENDPOINTS"] **Note:** Input list is expected for this operation. If the list is empty, the operation works similar to the ALL operation.

`DOWNLOAD`
 	
Admin CLI command:
 	
`collectappnodedata -p "D:/appNode/data/" -dp "D:/downloads" -al true -dd true DOWNLOAD`
	The operation is used to download the collected AppNode data.
	
	Available options:
	
	• `path`: [optional] the path for the directory where data is collected. OR the path for the file.
	• `all`: [optional] If the value of option "path" is a directory or if the value is not set, then the path is the default directory.
	
	If you set the "all" option as TRUE, all files present in that directory having names starting with a keyword "APPNODE_DATA" are compressed to a single zip file with the name "APPNODE_DATA", and then the file APPNODE_DATA.zip is downloaded.
	
	If the value is not set or set as false and the path value is a directory, then the last generated file is sent as output.
	• `downloadanddelete`: [optional] Delete the file after download. It has two options true or false.
	
	The default option is false.
	• `downloadpath`: [mandatory and applicable for CLI command] To provide download path directory.
	
	Output:
	
	The file is download at the specified download path.

Operation	Description
	Note: If an option `downloadanddelete` is selected and the file download operation fails because of network issue, the file is not available for download next time.

`LIST`
Admin CLI command:
`collectappnodedata -p "D:/appNode/data/" LIST`
	The operation is used to list the data file present at the set path.
	Available option:
	• `path`: [optional] the path for the directory where data is collected. OR the path for the file.
	Output:
	If the path set is a directory or is a default path, then all files present in that directory having names starting with a keyword "APPNODE_DATA" are listed as output. If the path is a file, then it checks if the file exists.

`DELETE`
Admin CLI command:
`collectappnodedata -p "D:/appNode/data/" DELETE`
	The operation is used to delete the data files created.
	Available option:
	• `path`: [optional] the path for the directory where data is collected. OR the path for the file.
	Output:
	If the path set is a directory or is a default path, then all files present in that directory having names starting with a keyword "APPNODE_DATA" are deleted. If the path is a file, then the file is deleted.
	Note: The files with names starting with the keyword "APPNODE_DATA" are deleted.

Admin UI

To collect an AppNode data of a running AppNode using Admin UI:

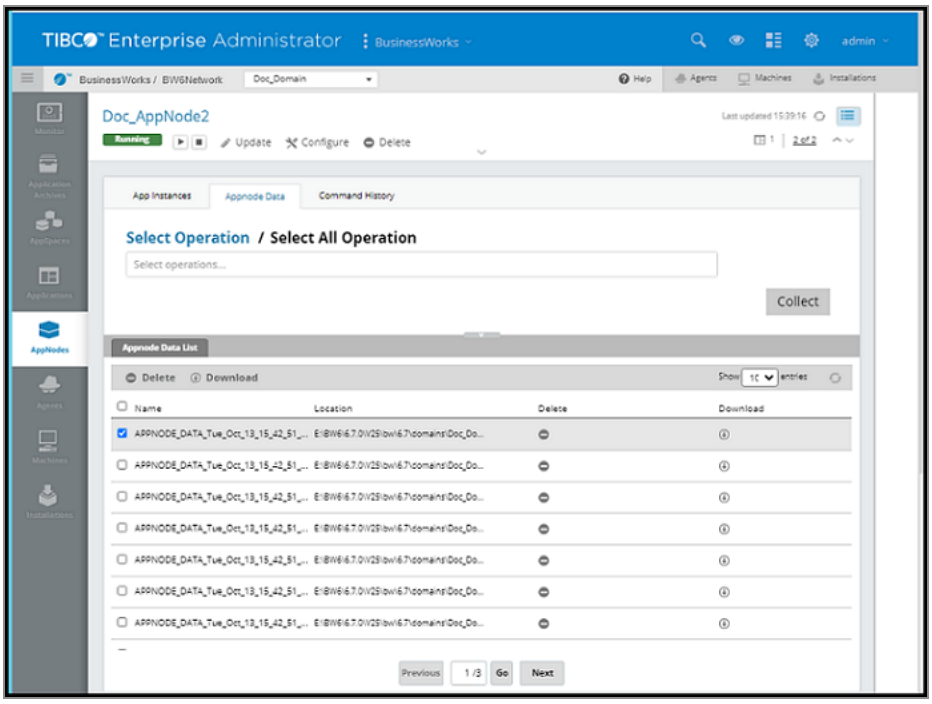
1. Navigate to the AppNode level 2 page.

2. Open the **Appnode Data** tab.
3. Select the operation from the list of default operations provided. By default, the following operations are provided:

   "THREAD_DUMP", "HEAP_DUMP", "VM_ARGUMENTS", "ENVIRONMENT_VARIABLES", "SYSTEM_PROPERTIES", "THREAD_SNAPSHOT", "MEMORY_SNAPSHOT","SYSTEM_PROCESS_INFORMATION","CPU_INFORMATION", "LMETRICS", "LCFG", "LP", "LA", "LENDPOINTS","LAPI"

   To select all operations from the list, select the **Select All Operation** option.

   To add the custom operation to the list, type a name of a custom operations and press Enter.
4. Click **Collect**. The dialog is displayed showing the list of operations selected. Clear the **Override** checkbox if you do not want to override the data file.
5. The AppNode data is collected at the `TIBCO_HOME/bw/<version>/domains/<domain_name>/appnodes/<AppSpace_Name>/<AppNode_Name>/debug` folder in the `.zip` format.
6. The **Appnode Data List** section shows the list of data files collected at the `TIBCO_HOME/bw/<version>/domains/<domain_name>/appnodes/<AppSpace_Name>/<AppNode_Name>/debug` folder on your file system.
   - To delete the data file from the **Appnode Data List** section as well as from your file system, click the **Delete** button at row level.
   - To delete multiple data files, select the checkboxes on the left side of those rows and click the **Delete** button on top of the list.
   - To download the AppNode's data on your local system, click the **Download** button on the right side of the data file in the list.
   - To download multiple data files, select the checkboxes on the left side of those rows and click the **Download** button on top of the list.
   - To select all entries irrespective of pagination, and perform bulk delete or download operations in the **AppNode Data List**, select the **Select All** checkbox, and click **Delete** or **Download** buttons on the top of the list.



REST API

API context	http://<host>:<port>/bw/framework.json/collect/{operation}
Method	POST
Authorization required	YES
Header-parameter	login
Operations	• `ALL` • `INCLUDE` • `EXCLUDE` • `DOWNLOAD` • `LIST` • `DELETE` For example: `http://<host>:<port>/bw/framework.json/collect/ALL`

The operation details are as follows:

Operation	Description
ALL	This API is used for running the default set of operations. The default set of operations is as follows: ["THREAD_DUMP", "HEAP_DUMP", "VM_ARGUMENTS","ENVIRONMENT_VARIABLES", "SYSTEM_PROPERTIES", "THREAD_SNAPSHOT","MEMORY_SNAPSHOT", "SYSTEM_PROCESS_INFORMATION","CPU_INFORMATION","LMETRICS", "LCFG", "LP", "LA", "LENDPOINTS","LAPI *"]
INCLUDE	This API accepts a list of commands or operations as an input in the form of a JSON list. Only listed operations are run.
EXCLUDE	This API accepts a list of commands or operations as an input in the form of a JSON list. All default set operations excluding the set of operation given as input is run.
DOWNLOAD	This API is available to download all collected data as a stream APPLICATION_OCTET_STREAM
LIST	This API is available to list the files present.
DELETE	This API is available to delete data files created.

Header Parameter	Description
PATH	An optional parameter to provide a directory path where the data is collected or is downloaded.
OVERRIDE	An option for collect data operation [ALL, INCLUDE, EXCLUDE], where the data collected previously is overwritten by the new data. The default value is TRUE.
ALL	An option for operation DOWNLOAD, where all files present are compressed at one file with the name "APPNODE_DATA.zip" and

Header Parameter	Description
	downloaded at once. The default value is FALSE.
DOWNLOADANDDELETE	An option for operation DOWNLOAD, where the file is deleted after the download operation. The default value is FALSE.
LOGIN	This is a mandatory parameter required for user authorization. It is the login id for the session.

API consumes entity: INPUT

Required Header parameter: Content-Type=application/json

JSON list of commands: Sample input: ["command1", "command2"]

Applicable for INCLUDE and EXCLUDE operations

Stopping an AppNode

When an AppNode is stopped, applications running on the AppNode stop.

BWAdmin Command Line

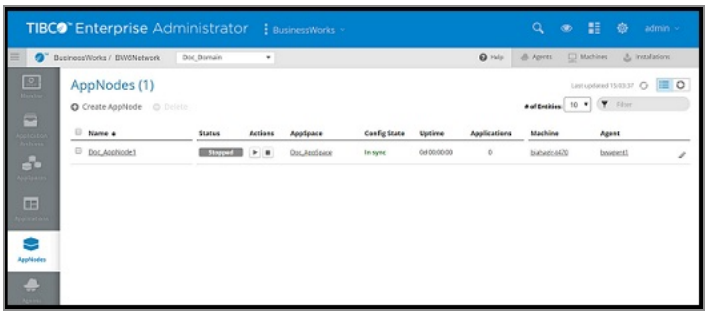
To stop the AppNode MyAppNode in the AppSpace MyAppSpace, run the following command at the command line:

```
BW_HOME\bin>bwadmin stop -d MyDomain -appspace MyAppSpace appnode
MyAppNode
```

Admin UI

On the **AppNode** page, click the **Stop** icon ■ for the AppNode.

The AppNode state changes from Running to Stopping, a transient state, then Stopped.



Force Shutting Down an AppNode

Use the argument, `-timeout` xx (in minutes) from the command line to forcefully shut down an AppNode, after the timeout is reached. The default timeout value is zero (0) and the AppNode stops only after the completion of all the jobs. From the Admin UI, select the **Force shutdown after wait time** checkbox. If the checkbox is not selected the AppNode stops after the default timeout. From the Admin UI, you can forcefully shut down AppNodes from the AppSpace level, the Application level, and from the Agent and Machine level.

When the timeout is specified, the AppNode is shut down after the timeout is reached. If the jobs are completed before the timeout value is reached, the AppNode stops on its own, and does not wait for the timeout period that has been specified. If the jobs are not completed in the timeout period, the AppNode is shut down irrespective of the state of the running jobs.

Note: Multiple force kill commands can be triggered one after the other, and the most recent force shut down command takes precedence over the previous commands.

BWAdmin Command Line

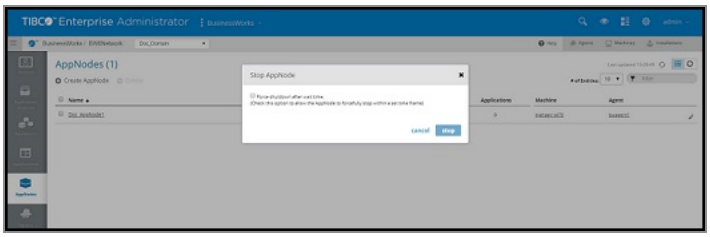
To force shut down an AppNode, `MyAppNode` in the AppSpace `MyAppSpace` run the following command at the command line:

```
BW_HOME\bin>bwadmin stop -timeout xx(time in minutes) -d MyDomain -
appspace MyAppSpace appnode MyAppNode
```

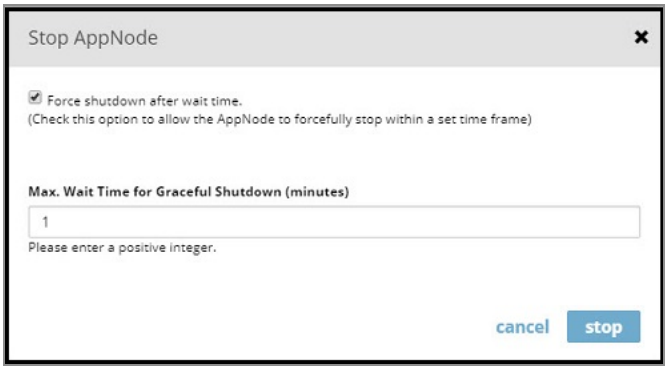
Admin UI

Procedure

1. On the **AppNode** page, click the **Stop** icon for the AppNode.



2. To shut down the AppNode forcefully within the timeout period, select the **Force shutdown after wait time** checkbox.
3. Set the timeout value (in minutes) in the **Max Wait Time for Graceful Shutdown** field and click the **stop** button.



Deleting an AppNode

Deleting an AppNode deletes any contained applications. From the BWAdmin command line, AppNodes that are running must be force deleted.

BWAdmin Command Line

To delete the AppNode `MyAppNode` in the AppSpace `MyAppSpace`, run the following command at the command line:

```
BW_HOME\bin>bwadmin delete -d MyDomain  -a MyAppSpace appnode MyAppNode
```

If the AppNode is running, the `delete appnode` command fails. You can stop the AppNode and retry the `delete appnode` command or use the `delete appnode` command with the `-force` option.

```
BW_HOME\bin>bwadmin delete -force -d MyDomain -a MyAppSpace appnode
MyAppNode
```

To delete the AppNode `MyAppNode` in the AppSpace `MyAppSpace` and forcefully shut down the running AppNode forcefully, run the following command at the command line:

```
BW_HOME\bin>bwadmin delete -timeout xx(time in minutes) -force -d
MyDomain -appspace MyAppSpace appnode MyAppNode
```

For more information, see Force Shutting Down an AppNode.

To delete all AppNodes in the AppSpace `MyAppSpace`, run the following command at the command line:

```
BW_HOME\bin>bwadmin delete -d MyDomain -a MyAppSpace -all appnode
```

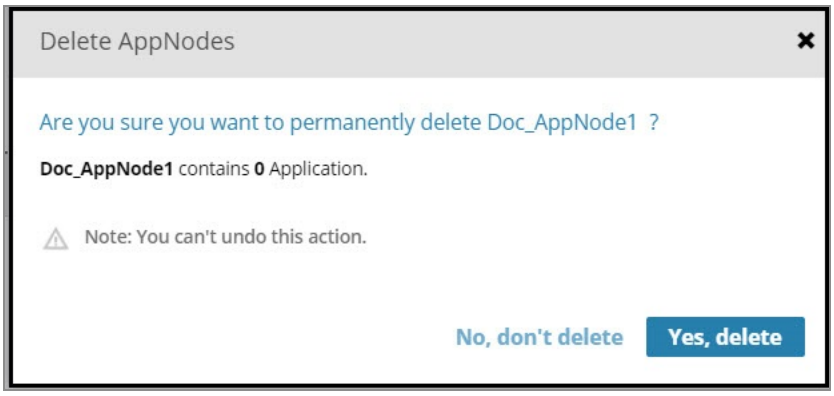
If any of the AppNode has running applications or if you want to delete all AppNodes forcefully, run the following command:

```
BW_HOME\bin>bwadmin delete -d MyDomain -a MyAppSpace -force -all appnode
```

Admin UI

Procedure

1. On the **AppNodes** page, click the check mark next to the AppNode to delete.
2. Click **Delete**.
3. Click **Yes, delete** in the **Delete AppNodes** dialog. The dialog message displays the number of applications that are deleted.)



Debugging an AppNode

A running AppNode can be enabled for remote debugging from either BWAdmin or the Admin UI. Once enabled, use TIBCO Business Studio for BusinessWorks to debug the application running on the AppNode. For more information, see "Remote Debugging" in the *TIBCO ActiveMatrix BusinessWorks™ Application Development* guide. An AppNode must be enabled for remote debugging in secure environments where only an administrator has the access rights to enable or disable ports.

BWAdmin Command Line

The `enabledebugport` command can only be run against a running AppNode. It should be issued from BWAdmin interactive mode, not from the command line.

Procedure

1. In a terminal, navigate to *BW_HOME*`\bin` and type bwadmin.
2. Go to **MyDomain**.

   ```
   bwadmin[admin]> cd MyDomain
   ```

3. Go to **MyAppSpace**.

   ```
   bwadmin[admin@MyDomain]> cd MyAppSpace
   ```

4. Start the AppNode, if it is not already running:

```
bwamdin[admin@MyDomain/MyAppSpace]> start appnode MyAppNode
```

5. Run the `enabledebugport` command, with the host and port number. For example:

```
bwamdin[admin@MyDomain/MyAppSpace]> enabledebugport -n MyAppNode
JSMITH-W520 9061
Enabled debug port on AppNode [MyAppNode] in AppSpace [MyAppSpace]
in Domain [MyDomain]
```

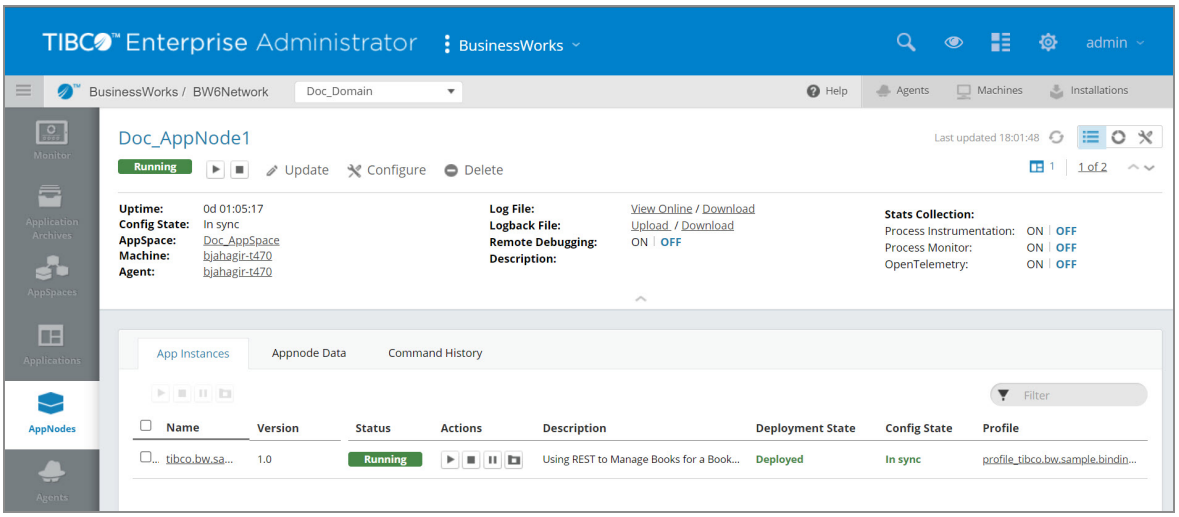
6. **Important:** When you finish debugging, close the port to reduce security risks and reduce overhead. For example:

```
bwamdin[admin@MyDomain/MyAppSpace]> disabledebugport -n MyAppNode
Debugger disabled for AppNode [MyAppNode] in AppSpace [MyAppSpace]
in Domain [MyDomain]
```

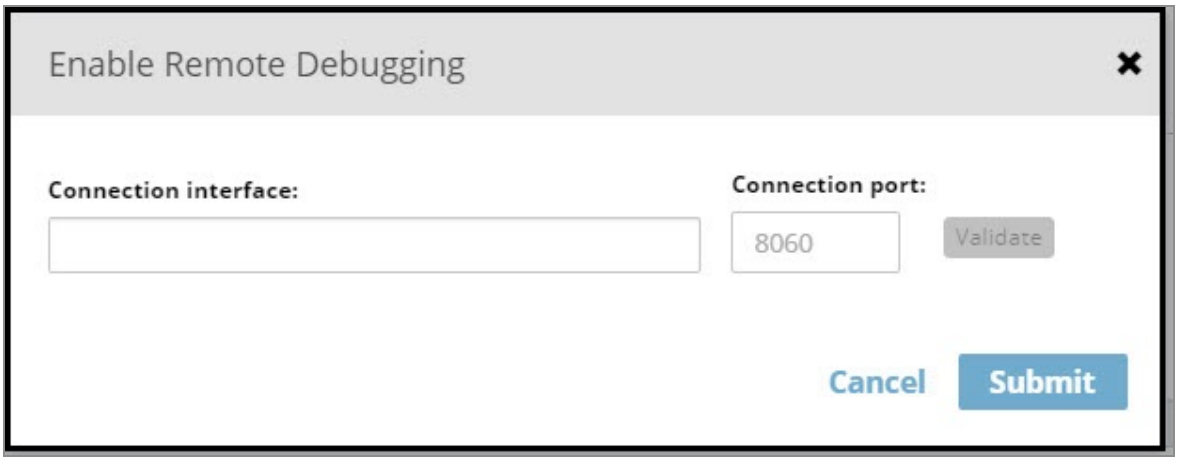
Admin UI

Procedure

1. Open the **AppNode** page for the AppNode to enable for remote debugging.



2. Click **Remote Debugging > ON**. The Enable Remote Debugging dialog is displayed.



3. Enter the following information, and click **Submit** to open the port.
   - **Connection Interface**: The default connection interface is the name of the BWAgent.
   - **Connection Port**: The debug port.

   The port is opened and displayed on the AppNode page:

4. When you finish debugging, close the port by clicking **OFF** to reduce security risks and reduce overhead.

OSGi Commands

You can run commands to gather data about running AppNodes and applications.

Command Reference

- To view all commands, use

```
curl -v http://localhost:8090/bw/framework.json/osgi?command=help
```

- To view command syntax, use

```
curl -v
http://localhost:8090/bw/framework.json/osgi?command=help%20<comman
d_name>
```

For example,

```
curl -v
http://localhost:8090/bw/framework.json/osgi?command=help%20pauseap
p
```

The following table lists some of the commands.

OSGi Commands

Command	Description	Options	Scope	Parameters	Flags	Usage
`dsr`	Diagnoses Shared Resource issues.	-	`bw`	String: Partial or full name of a Shared Resource. Case is ignored.	-	`osgi "dsr"` OR `osgi "dsr <id>"` OR `osgi "dsr <name>"`
`geticon`	• Test for availability of all	-	`bw`	• String	-	For activity type, use `osgi "lat"` command. `osgi "geticon activityid"`

Command	Description	Options	Scope	Parameters	Flags	Usage
	the ActiveMatrix BusinessWorks Activity Icons with a given ID. • Test for availability of an ActiveMatrix BusinessWorks Activity Icon with a given ID and type.			• String: id String: type		

Command	Description	Options	Scope	Parameters	Flags	Usage
	• Test for availability of an ActiveMatrix BusinessWorks Activity Icon.					
`lais`	Retrieves statistics of activities that have been run in one or more processes for the application.	-p, --process Name of the process. [optional] -v, --version Version of the application. [optional]	`bw`	String: Name of the application	-	`osgi "lais continuousTimer.application"`

Command	Description	Options	Scope	Parameters	Flags	Usage
`lapi`	Retrieves information about process instances of the application.	-o, --output File path to export output in CSV format. [optional] -p, --process Name of the process. [optional] -s, --state State of the process instance to be filtere	`bw`	String: Name of the application (Use * instead of application name to list process instances of all the applications)	-	`osgi "lapi continuousTimer.application"`

Command	Description	Options	Scope	Parameters	Flags	Usage
		d. (Currently the possible values to filter based on process instance state are [SUSPENDED, ACTIVE, ALL].) [optional] -v, --version Version of the application. [optional]				

Command	Description	Options	Scope	Parameters	Flags	Usage
`lapis`	Prints summary of active process instances.	-	bw	String: Process Instance ID	-	`osgi "lapis 1ef60fb4-29b-4db1-b0ec-cdce681254ee"`
`las`	Lists all instantiated activities.	-	bw	-	-	`osgi "las"`
`lat`	Lists all registered activity types.	-	bw	-	-	`osgi "lat"`
`lbwes`	Lists all subscribers that are currently listening to ActiveMatrix BusinessWorks statistics events.	-	bw	-	-	`osgi "lbwes"`
`le`	Prints information about ActiveMatrix BusinessWorks engines.	-	bw	-	-	`osgi "le"`
`lec`	Prints information about	-	bw	-	-	`osgi "lec"`

Command	Description	Options	Scope	Parameters	Flags	Usage
	ActiveMatrix BusinessWorks engine configurations.					
`lendpoints`	Lists endpoints exposed by the ActiveMatrix BusinessWorks engine.	-a, --app Name of the application. [optional] -t, --type Endpoint Type [optional]	`bw`	-	-	`osgi "lendpoints"`
`les`	Lists all instantiated `EventSources`.	-	`bw`	-	-	`osgi "les"`
`lmetrics`	Prints job metrics for one or more applications running on the AppNode.	-a, --app Name of the application. [optional]	`bw`	-	-	`osgi "lmetrics"`

Command	Description	Options	Scope	Parameters	Flags	Usage
		-v, --version Version of the application. [optional]				
lpis	Prints statistics of one or more processes that have been run for the application.	-p, --process Name of the process [optional] -v, --version Version of the application [optional]	bw	String: Name of the application	-	osgi "lpis continuousTimer.application"
lr	• Lists all resou	long Shared	bw	-	-	osgi "lr"

Command	Description	Options	Scope	Parameters	Flags	Usage
	rces. • Lists all resource details.	Resource ID				
`lrhandlers`	Lists all resource handlers.	-	`bw`	-	-	`osgi "lrhandlers"`
`lrproxies`	Lists all resource proxies.	-	`bw`	-	-	`osgi "lrproxies"`
`startesc`	Starts collection of execution statistics for a given entity (activity/process) for one or more applications.	-a, --app Name of the application [optional] -v, --version Version of the application [optional]	`bw`	String: (Activity Name/ Process Name)	-	`osgi "startesc activity | startesc Process"`

Command	Description	Options	Scope	Parameters	Flags	Usage
`stopesc`	Stops collection of execution statistics for a given entity (activity/process) for one or more applications.	-a, --app Name of the application [optional] -v, --version Version of the application [optional]	`bw`	String: (Activity Name/ Process Name)	-	`osgi "stopesc activity | stopesc Process"`
`startpsc`	Starts collection of process statistics for one or more applications.	-a, --app Name of the application [optional] -v, --version Version of the applic	`bw`	-	-	`osgi "startpsc"`

Command	Description	Options	Scope	Parameters	Flags	Usage
		ation [optional]				
`stoppsc`	Stops collection of process statistics for one or more applications.	-a, --app Name of the application [optional] -v, --version Version of the application [optional]	`bw`	-	-	`osgi "stoppsc"`
`appnodeprocessinfo`	Prints information about the AppNode system processes.	-	`fr wk`	-	-	`osgi "appnodeprocessinfo"`
`dc`	Delete a configuration with a given PID.	-	`fr wk`	String: PID of the configuration	-	`osgi "dc <process id>"`

Command	Description	Options	Scope	Parameters	Flags	Usage
`getcompdetail`	Get the detail of a specific component of an ActiveMatrix BusinessWorks application.	-v, --version Version of the application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application String: Name of the Component	-	`osgi "getcompdetail tibco.bw.sample.binding.rest.Basic.application Componentbooks"`
`getcomps`	Get the list of components of an ActiveMatrix BusinessWorks application.	-v, --version Version of the Application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application	-	`osgi "getcomps tibco.bw.sample.binding.rest.Basic.application"`

Command	Description	Options	Scope	Parameters	Flags	Usage
`la`	Print information about all applications.	-	`frwk`	-	-	`osgi "la"`
`lap`	Print all application properties.	-	`frwk`	-	-	`osgi "lap"`
`lb`	• List all installed bundles. • List installed bundles matching a substring.	-	`frwk`	String: Substring matched against name or symbolic name	• `-l, --location` show location • `-s, --symbolicname` show symbolic name • `-u, --update`	`osgi "lb"`

Command	Description	Options	Scope	Parameters	Flags	Usage
					`location` show update location	
`lcfg`	Print all CAS configuration details.	-	`fwk`	String: Filter expression	-	`osgi "lcfg"`
`lp`	Print information about all known ActiveMatrix BusinessWorks processes.	-	`fwk`	-	-	`osgi "lp"`
`ll`	Print information about all libraries.	-	`fwk`	-	-	`osgi "ll"`
`lloggers`	Print all loggers currently configured on the AppNode.	-	`fwk`	-	-	`osgi "lloggers"`

Command	Description	Options	Scope	Parameters	Flags	Usage
`pauseapp`	Stop the process starters and their bindings. Also pause all jobs of the ActiveMatrix BusinessWorks application.	-v, --version Version of the application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application	-	`osgi "pauseapp tibco.bw.sample.binding.rest.Basic.application"`
`resumeapp`	Start the process starters and their bindings. Also resume all jobs of the ActiveMatrix BusinessWorks application.	-v, --version Version of the application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application	-	`osgi "resumeapp tibco.bw.sample.binding.rest.Basic.application"`

Command	Description	Options	Scope	Parameters	Flags	Usage
`resumepi`	Resumes the process instances for specified process instance ids.	-	`fr wk`	String: Active Matrix BusinessWorks generated process instance id or list of ids separated by comma. For example, `resumepi p_id1,p_id2`.	-	`osgi "resumepi p_id1,p_id2"`
`setloglevel`	Sets the log level for a given logger.	-l, --level Log level to set. The valid values are {Trace, Debu	`fr wk`	String: Name of the Logger	-	`osgi "setloglevel ROOT -l DEBUG"`

Command	Description	Options	Scope	Parameters	Flags	Usage
		g, Info, Error, Warn}. If not provided, log level is set to parent log level. [optional]				
`startcomps`	Start all process starters and their bindings of the ActiveMatrix BusinessWorks application.	-v, --version Version of the application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application	-	`osgi "startcomps tibco.bw.sample.binding.rest.Basic.application"`

Command	Description	Options	Scope	Parameters	Flags	Usage
`startps`	Start the process starters of the ActiveMatrix BusinessWorks application.	-v, --version Version of the Application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application	-	`osgi "startps tibco.bw.sample.binding.rest.Basic.application"`
`stopps`	Stop the process starters of the ActiveMatrix BusinessWorks application.	-v, --version Version of the Application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application	-	`osgi "stopps tibco.bw.sample.binding.rest.Basic.application"`

Command	Description	Options	Scope	Parameters	Flags	Usage
`startapp`	Start an ActiveMatrix BusinessWorks application gracefully.	-v, --version Version of the Application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application	-	`osgi "start tibco.bw.sample.binding.rest.Basic.application"`
`stopapp`	Stop an ActiveMatrix BusinessWorks application gracefully.	-v, --version Version of the Application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application	-g, --graceful If set causes the application to shut down gracefully.	`osgi "stopapp tibco.bw.sample.binding.rest.Basic.application"`

Command	Description	Options	Scope	Parameters	Flags	Usage
`startcomp`	Start the specific component of an ActiveMatrix BusinessWorks application.	-v, --version Version of the Application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application String: Name of the Component	-	`osgi "startcomp tibco.bw.sample.binding.rest.Basic.application Componentbooks"`
`stopcomp`	Stop the specific component of an ActiveMatrix BusinessWorks application.	-v, --version Version of the Application. If omitted, all versions are affected. [optional]	`frwk`	String: Name of the Application String: Name of the Component	-	`osgi "stopcomp tibco.bw.sample.binding.rest.Basic.application Componentbooks"`

Command	Description	Options	Scope	Parameters	Flags	Usage
`suspendpi`	Suspends the process instances for specified process instance ids.	-	`fr wk`	String: Active Matrix BusinessWorks generated process instance id or list of ids separated by comma. For example, `suspendpi p_id1,p_id2`.	-	`osgi "suspendpi p_id1,p_id2"`
`td`	Print a full thread dump.	-	`fr wk`	-	-	`osgi "td"`
`type`	-	-	`go go`	String []	-	-
`clean`	Removes caches for the bundle with the given id.	-	`ws dl`	String	-	`wsdl:clean 5`

Command	Description	Options	Scope	Parameters	Flags	Usage
clean	Removes caches for all known bundles.	-	wsdl	-	-	wsdl:clean
collisions	Lists collisions for given bundle id.	-	wsdl	String	-	wsdl:collisions 5
collisions	Lists collisions across all projects.	-	wsdl	-	-	wsdl:collisions
elem	Locates element declarations with the given local name across all projects.	-	wsdl	String	-	wsdl:elem testname
elem	Locates element declarations across all projects.	-	wsdl	-	-	wsdl:elem
elemLoc	Locates element declarations with the schemaCom	-	wsdl	String	-	wsdl:elemLoc location

Command	Description	Options	Scope	Parameters	Flags	Usage
	ponentCache for the WSDL at the provided location.					
errors	Prints error messages for specified bundle id.	-	wsdl	long	-	wsdl:errors 7
errors	Prints error messages for all bundles.	-	wsdl	-	-	wsdl:errors
ids	Prints bundle IDs of all bundles with component caches.	-	wsdl	-	-	wsdl:ids
imports	Shows basic info such as locations, targetNamespaces, and imports.	-	wsdl	-	-	wsdl:imports
imports	Shows information related to imports for	-	wsdl	long	-	wsdl:imports 12

Command	Description	Options	Scope	Parameters	Flags	Usage
	a given bundle ID.					
`load`	Loads resources for all known bundles.	-	`wsdl`	-	-	`wsdl:load`
`load`	Loads resources for the bundle with the given ID.	-	`wsdl`	String	-	`wsdl:load 1`
`locations`	Lists the locations for a given namespace.	-	`wsdl`	String	-	`wsdl:locations namespace`
`locations`	Prints the locations across all bundles.	-	`wsdl`	-	-	`wsdl:locations`
`ns`	Prints the namespaces for the given bundle ID.	-	`wsdl`	String	-	`wsdl:ns 12`
`ns`	Prints namespaces for all	-	`wsdl`	-	-	`wsdl:ns`

Command	Description	Options	Scope	Parameters	Flags	Usage
	bundles.					
nsLoc	Prints "ns" in the ModuleCache for each WSDL location.	-	wsdl	-	-	wsdl:nsLoc
type	Locates type definitions with the given local name across all projects.	-	wsdl	String	-	wsdl:type testlocalname
types	Lists all type definitions for a specified project.	-	wsdl	long	-	wsdl:types 5
types	Lists all type definitions across all projects.	-	wsdl	-	-	wsdl:types
typeLoc	Prints type declarations with the schemaCom	-	wsdl	String	-	wsdl:typeLoc location

Command	Description	Options	Scope	Parameters	Flags	Usage
	ponentCache for the WSDL at the provided location.					
`wires`	Prints cache dependencies (as bundle IDs) for the given bundle ID.	-	`wsdl`	String	-	`xsd:wires 5`
`att`	Locates attribute declarations with the given local name across all projects.	-	`xsd`	String	-	`xsd:att testlocalname`
`atts`	Lists all attribute declarations for a specified bundle.	-	`xsd`	`long`	-	`xsd:atts 5`
`atts`	Lists all attribute declarations across all projects.	-	`xsd`	-	-	`xsd:atts`

Command	Description	Options	Scope	Parameters	Flags	Usage
clean	Removes caches for the bundle with the given ID.	-	xsd	String	-	xsd:string 5
clean	Removes caches for all known bundles.	-	xsd	-	-	xsd:clean
collisions	Lists collisions for a given bundle ID.	-	xsd	String	-	xsd:collisions 5
collisions	Lists collisions across all projects.	-	xsd	-	-	xsd:collisions
elem	Locates element declarations with the given local name across all projects.	-	xsd	-	-	xsd:elem testname
elem	Locates element declarations with the given local	-	xsd	String	-	xsd:elem

Command	Description	Options	Scope	Parameters	Flags	Usage
	name across all projects.					
`elemLoc`	Locates element declarations with the schemaComponentCache for the schema at the provided location.	-	`xsd`	String	-	`wsdl:elemLoc location`
`elemLoc`	Locates element declarations with the schemaComponentCache for the schema at the provided location.	-	`xsd`	String String	-	`xsd:elemLoc location elemName`
`elemNs`	Locates element declarations with the given targetNamespace.	-	`xsd`	String	-	`xsd:elemNs namespace`

Command	Description	Options	Scope	Parameters	Flags	Usage
`elemNsEmpty`	Locates element declarations with empty string targetNs.	-	`xsd`	-	-	`xsd:elemNsEmpty`
`elemNsNull`	Locates element declarations with null targetNs.	-	`xsd`	-	-	`xsd:elemNsNull`
`elemsNullLocEmptyNs`	Locates all element declarations with the null location and empty "`tns`".	-	`xsd`	-	-	`xsd:elemsNullLocEmptyNs`
`elemsNullLocNullNs`	Locates all element declarations with the null location and null "`tns`".	-	`xsd`	-	-	`xsd:elemsNullLocNullNs`
`errors`	Prints error messages for a specified	-	`xsd`	`long`	-	`xsd:errors 5`

Command	Description	Options	Scope	Parameters	Flags	Usage
	bundle ID.					
errors	Prints error messages for all the bundles.	-	xsd	-	-	xsd:errors
errorsLoc	Prints error messages for a specified location.	-	xsd	String	-	xsd:errorsLoc location
errorsNs	Prints error messages for a specified namespace.	-	xsd	String	-	xsd:errorNs namespace
ids	Prints the bundle IDs of all the bundles with component caches.	-	xsd	-	-	xsd:ids
imports	Shows basic information such as locations, targetNamespaces, and imports.	-	xsd	-	-	xsd:imports

Command	Description	Options	Scope	Parameters	Flags	Usage
`imports`	Shows information related to imports for a given bundle ID.	-	`xsd`	`long`	-	`xsd:imports 12`
`load`	Loads resources for all known bundles.	-	`xsd`	-	-	`xsd:load`
`load`	Loads resources for the bundle with the given ID.	-	`xsd`	String	-	`xsd:load 1`
`locations`	Prints XSD locations across all bundles.	-	`xsd`	-	-	`xsd:locations`
`locations`	Lists locations for a given namespace.	-	`xsd`	String	-	`xsd:locations namespace`
`mg`	Locates specific named model group	-	`xsd`	String	-	`xsd:mg name`

Command	Description	Options	Scope	Parameters	Flags	Usage
	declaration.					
mg	Locates all named model group declarations.	-	xsd	-	-	xsd:mg
ns	Prints namespaces for the given bundle ID.	-	xsd	String	-	xsd:ns 12
ns	Prints namespaces for all bundles.	-	xsd	-	-	xsd:ns
nsLoc	Prints namespaces in the ModuleCache for each WSDL location.	-	xsd	-	-	xsd:nsLoc
type	Locates type definitions with the given local name across all	-	xsd	String	-	xsd:type testlocalname

Command	Description	Options	Scope	Parameters	Flags	Usage
	projects.					
`types`	Lists all type definitions for specified projects.	-	`xsd`	`long`	-	`xsd:types 30`
`types`	Lists all type definitions across all projects.	-	`xsd`	-	-	`xsd:types`
`typeLoc`	Prints type declarations with the schemaComponentCache for the XSD at the provided location.	-	`xsd`	String	-	`xsd:typeLoc location`
`wires`	Prints cache dependencies (as bundle IDs) for all bundles.	-	`xsd`	-	-	`xsd:wires`

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

Backing Up and Restoring an AppNode

Backing up an AppNode exports the current state of the specified AppNode to a BWAdmin command file. The command file can be provided to BWAdmin to recreate the AppNode. Output can be compressed to a ZIP file with the `-zipped` option.

Procedure

1. To back up the current state of an AppNode,
    a. Open a terminal and navigate to *BW_HOME*`\bin`.
    b. Enter the `backup` command from the command line, using `-s` option to identify the name of the destination file. Use the `-domain` and `-appspace` options, with the `appnode` argument in the command line. The AppNode can be either a local AppNode or an AppNode in a BWAgent in the agent network. By default, destination files are written to the current working directory.

       This example backs up AppNode `MyAppNode` in a BWAgent network to a command file named `MyAppNode.cmd`.

       ```
       BW_HOME\bin>bwadmin backup -s MyAppnode.cmd -d Machine2Domain -
       a AS1 appnode MyAppNode
       ```

2. To restore the AppNode,
    a. Open a terminal and navigate to *BW_HOME*`\bin`.
    b. Enter the `bwadmin` command, providing the name of the backup command file. The following example recreates the AppNode `MyAppNode`.

       ```
       BW_HOME\bin>bwadmin -f MyAppnode.cmd
       ```

       If you are restoring to a different location, you need to update the command file as follows:

       - The agent name points to `localhost` by default. You need to change this to the name of the machine that you are restoring to.
       - Update the domain home to point to the absolute path to the new location.
       - Update the path to the application archive (EAR) file to an absolute path.

    c. Use the `bwadmin show appnodes` command at the command line with the –

`domain` and `-appspace` options to verify the restore.

Note: Always use the available port while creating any AppNode, both from the BW_HOME scope and system scope. In case of restoration, if the designated AppNode port is not available, change the port manually.

Restoring the File System of an AppNode

Restoring an AppNode restores the file system of the specified AppNode and all runtime entities in the AppNode to the state of the datastore.

Before you begin

- The names of the containing domain and AppSpace and the name of the AppNode must be known in order to restore.
- The BWAgent must be running.

Procedure

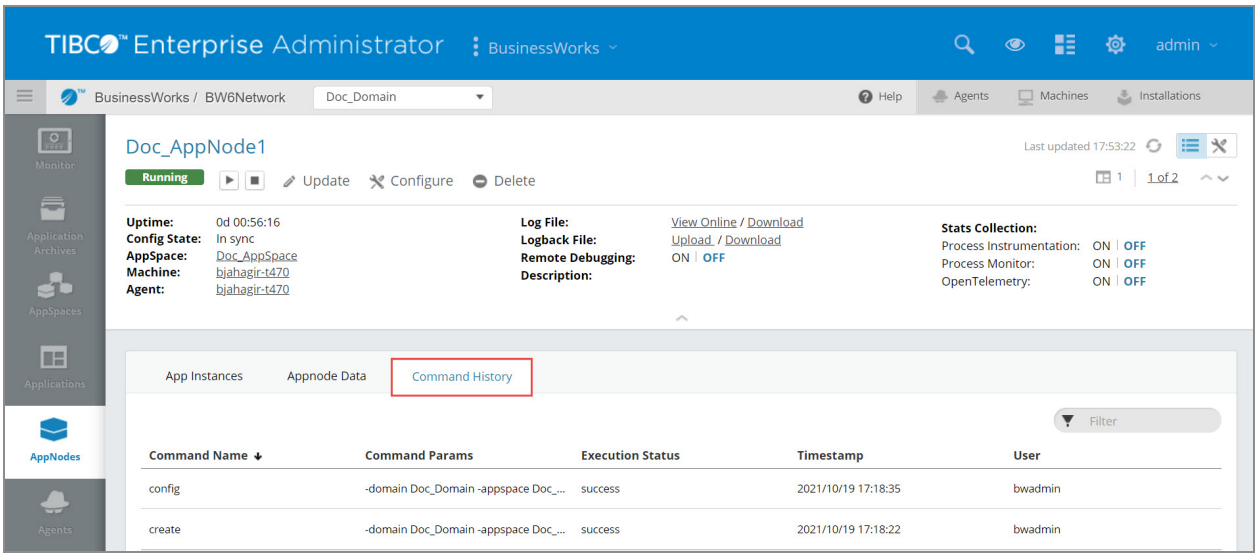
1. To restore the file system for an AppNode and the runtime entities in the AppNode, open a terminal and navigate to *BW_HOME*`\bin`.
2. Enter the `restore` command, using the `-domain` and `-appspace` options with the `appnode` argument specifying the name of the AppNode to restore. This example restores AppNode `MyAppNode` in domain `Machine1Domain` and AppSpace `AS1`.

   ```
   BW_HOME\bin>bwadmin restore -d Machine1Domain -a AS1 appnode
   MyAppNode
   ```

3. To verify the restore, check the file system. Open the *BW_HOME*`\domains` folder. Browse the folder and look for the named AppNode folder under: *BW_HOME*`\domains\`*domain_name*`\appnodes`

Command History

Open the **Command History** tab to view the commands or operations that were performed on an AppNode.



AppNode-Level Engine Properties

List of properties

Property	Description
`bw.engine.enable.loop.reset=true`	To enable the reset variables used in the loops.
`bw.independent.component.startup=true`	To run the application even when one of the components fails due to an incorrect configuration of the shared resource. The rest of the components can be started after using this property.
`bw.engine.enable.checkpointdata.delete=true`	To clear the data from the `pg_largeobject` table.
`bw.process.deserialization.in.parallel=true`	To reduce the deserialized time of ActiveMatrix BusinessWorks applications when set to "true". It has a major impact on applications having a large number of processes.
`bw.frwk.version.format`	To print the `major.minor.patch` application versions in logs. The version format can be set using a system

Property	Description
	property in the following formats: • `bw.frwk.version.format=major.minor` • `bw.frwk.version.format=major.minor.micro` • `bw.frwk.version.format=major.minor.micro.qualifier`

Enabling the OSGi Console for an AppNode

The `enableconsole` command dynamically enables the OSGi console on the given port for a running AppNode. Advanced users can use `telnet` command to the port and run native OSGi commands to get information about an AppNode's status. This is useful when collecting diagnostic data remotely. By default, the OSGi port is closed.

Note: Although an AppNode can be created with OSGi port details specified, this is not recommended. Keeping this port open when the console is not in use poses a security risk.

The `enableconsole` command can only be run against a running AppNode. It should be issued from BWAdmin interactive mode, not from the command line.

If you are testing and running applications in TIBCO Business Studio for BusinessWorks, you can also access the OSGi commands from the **Console view**.

BWAdmin Command Line

Procedure

1. In a terminal, navigate to *BW_HOME*`\bin` and type `bwadmin`.
2. Go to **MyDomain**.

```
bwadmin[admin]> cd MyDomain
```

3. Go to **MyAppSpace**.

```
bwadmin[admin@MyDomain]> cd MyAppSpace
```

4. Start the AppNode, if it is not already running:

```
bwamdin[admin@MyDomain/MyAppSpace]> start appnode MyAppNode
```

5. Run the `enableconsole` command, using the host and OSGi port number. For example:

```
bwamdin[admin@MyDomain/MyAppSpace]> enableconsole -n MyAppNode
JSMITH-W520 9060
TIBCO-BW-ADMIN-CLI-300304: Console enabled for AppNode [MyAppNode]
in Domain [MyDomain]
```

> **Note:** You can also specify the OSGi port number using one of the following syntaxes:
>
> - `hostname:`*`port number`*
> - `localhost:`*`port number`*

6. Open a new terminal window and use the `telnet` command to access the OSGi console:

   ```
   telnet JSMITH-W520 9060
   ```

   The OSGi console is opened in a terminal.

7. Use OSGi commands to retrieve information about the engine, the AppNode, the running application. For a list of commands, enter `help`. For more information, see OSGi Commands.

8. Use the `disconnect` command to quit the telnet session, and leave the OSGi port open for reentry. Use the `telnet stop` command to close the connection after the debugging session is complete. Do not use the `telnet exit` command as this shuts down the AppNode.

Running OSGi Commands

You can run OSGi commands from the Admin CLI, SSH Client, and HTTP Client.

- Running OSGi Commands from BWAdmin Command Line
- Running OSGi Commands Using SSH Client
- Running OSGi Commands Using HTTP Client

Running OSGi Commands from BWAdmin Command Line

Follow these steps to run OSGI commands from the BWAdmin command line.

Before you begin

Ensure the AppNode is running.

Procedure

1. From the CLI, navigate to the `/bin` folder.
2. Start BWAdmin.
3. Enter an OSGI command. OSGI commands can be run from the command line in the format `osgi [options] [command]`. In the following example, the option `-n` is used to specify the AppNode `MyAppNode`, and the `la` command is used to print information about applications on `MyAppNode`.

```
bwamdin[admin@MyDomain/MyAppSpace]> osgi -n MyAppNode "la"
```

OSGI Command Options

Option	Description
`-descr/ -description`	Description of an entity
`-t/-outputfile`	The OSGI command output is provided in a newly created text file. The text file is written to the `/bin` folder by default, but you can specify a different location, for example, `C:/temp/OSGI_OUTPUT.txt`.
`-d/-domain`	Specifies the domain name.

Option	Description
-a/-appspace	Specifies the name of the AppSpace.
-n/-appnode	Specifies the name of the AppNode.
--help	Lists all OSGI commands. For more information about specific commands, see OSGi Commands.

Running OSGi Commands Using SSH Client

You can run OSGi commands using SSH client.

Procedure

1. Edit the AppNode `config.ini` file.
2. Uncomment the following properties:

```
#osgi.console=null
#osgi.console.ssh=<free port>
#osgi.console.enable.builtin=false
#osgi.console.ssh.useDefaultSecureStorage=true

#java.security.auth.login.config=../sshconfig/equinox.console.jaas.
login.conf
#ssh.server.keystore=../sshconfig/hostkey.ser
#org.eclipse.equinox.console.jaas.file=../sshconfig/store
```

3. Configure the property `osgi.console` with the host name.
4. Configure the property `osgi.console.ssh` with the free port. This port is used for OSGi remote access.

   Default credentials to connect via SSH are `equinox` and `equinox`
5. Restart the AppNode.

Running OSGi Commands Using HTTP Client

You can run OSGi commands using HTTP client. The recommended approach is to use `curl`.

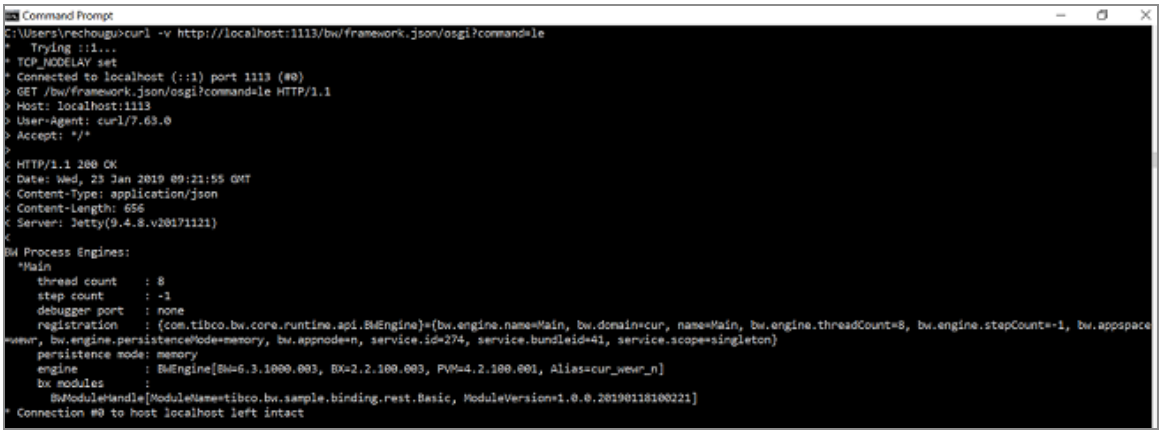
Run the following OSGi command using `curl` in the following format:

```
curl -v http://<IP address>:<port on which AppNode is
running>/bw/framework.json/osgi?command=<OSGi command>
```

For example:

- To print information about BWEngines:

  ```
  curl -v http://localhost:2224/bw/framework.json/osgi?command=le
  ```

- To pause all jobs of TIBCO ActiveMatrix BusinessWorks™ applications:

  ```
  curl -v
  http://localhost:1113/bw/framework.json/osgi?command=pauseapp%20-
  v%201.0%20tibco.bw.sample.binding.rest.BookStore.application
  ```

Disabling OSGi Commands

To disable OSGi commands, use the `bw.osgi.disable` property as an environment variable.

When the property `bw.osgi.disable` is set to "true", it disables the OSGi commands on the port and you can still start the AppNode. But it does not accept any OSGi requests.

Managing an Application

An application is an instance of a deployment archive. It has an independent lifecycle with regard to an AppSpace or AppNode. After completing the design process in TIBCO Business Studio, the application can be deployed.

An application provides the business logic to perform one or more related tasks and contains an application module that was defined in TIBCO Business Studio. The module itself can include processes, subprocesses, a process starter or a process service, and multiple activities. See *TIBCO ActiveMatrix BusinessWorks Application Development* for information about creating applications in TIBCO Business Studio.

Deployment occurs after an archive is uploaded to a domain and before an application is started. An archive is deployed to an AppSpace. One or more applications can be deployed to an AppSpace.

If an AppSpace spans multiple machines, the application is deployed onto each of the machines. If there are multiple AppNodes attached to the AppSpace either on a single machine or across multiple machines, the `start` command starts the application on each of the AppNodes. The runtime status of the application is reported for each AppNode and can be monitored using BWAdmin.

When an application is deployed, you can also choose to start the application by giving the `-as` option on the command line. By default, this option is off and the application must be started explicitly after being deployed.

To configure an application, provide the desired profile that should contain the variable values for the application. This step is necessary if you want to run the application with different sets of variables and deploy it with different argument values, for example, for a Windows machine or a Mac.

Preparing for Deployment

Preparation for deployment involves the following steps:

- Creating an Application
- Creating an Application with Multiple Profiles
- Creating an Archive

After creating the application, the profile, and the archive, you are ready to deploy the application into an AppSpace. Deploying an application involves the following steps:

- Uploading an Archive
- Deploying an Archive
- Configuring an Application
- Starting an Application

Creating an Application

This section shows how to create a simple project where you design and create an application.

After creating the project, you choose activities from the palettes to design and create an application. For more information, see *TIBCO ActiveMatrix BusinessWorks™ Application Development*.

Procedure

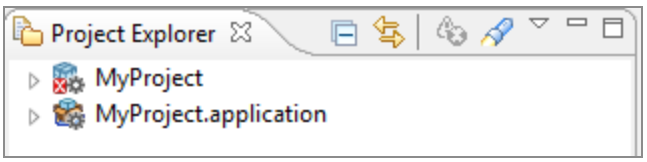
1. Start TIBCO Business Studio™ for BusinessWorks™.
2. Open the BusinessWorks Application Module wizard by selecting **File > New > Project > BusinessWorks > BusinessWorks Application Module** and click **Next**.
3. In the **Project name** field, provide a project name.

   Select the **Use default location**, **Create empty process**, and **Create Application** checkboxes.
4. Click **Finish**.

Result

The new project is visible in the **Project Explorer**.

New Project in Project Explorer



Creating an Application with Multiple Profiles

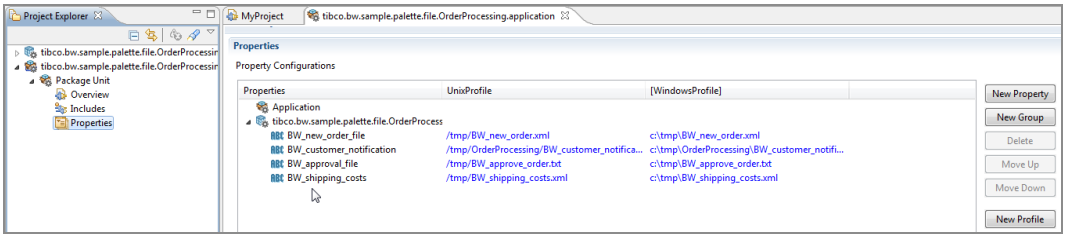
You can define multiple profiles when creating an application in TIBCO Business Studio™ for BusinessWorks™.

A *profile* is a collection of module and application properties that an application uses. When an application is deployed with different properties, different profiles are available for each deployment. For example, you can create a Windows profile for an application that runs on a Windows machine and another for the same application running on a UNIX machine.

Before you begin

An application is created with profiles using TIBCO Business Studio for BusinessWorks. For more information about creating applications, see the *TIBCO ActiveMatrix BusinessWorks™ Application Development* guide. The following screenshot shows an application with a profile for Windows and another for UNIX. Each profile has a set of defined properties and values. The values use the appropriate operating system syntax to point to the files in the file system. The files are created and maintained outside of TIBCO Business Studio for BusinessWorks.

Application Profiles



Follow these steps to create an application profile:

Procedure

1. Start TIBCO Business Studio for BusinessWorks and open an application.
2. Expand the application and double-click **Properties** under **Package Unit**.

   This displays the **Properties** pane in the **Process Editor**.
3. Click the **New Profile** button to add a new profile.
4. In the **Create New Profile** window, enter a name for the new profile. For example,

enter `WindowsProfile` and click **OK**.

The WindowsProfile is created and is available on the right of the **[default]** column in the Properties pane.

5. Double-click the field under the profile that corresponds to a property, and enter a value for the property.
6. Save the project.

   You can create multiple profiles as needed.

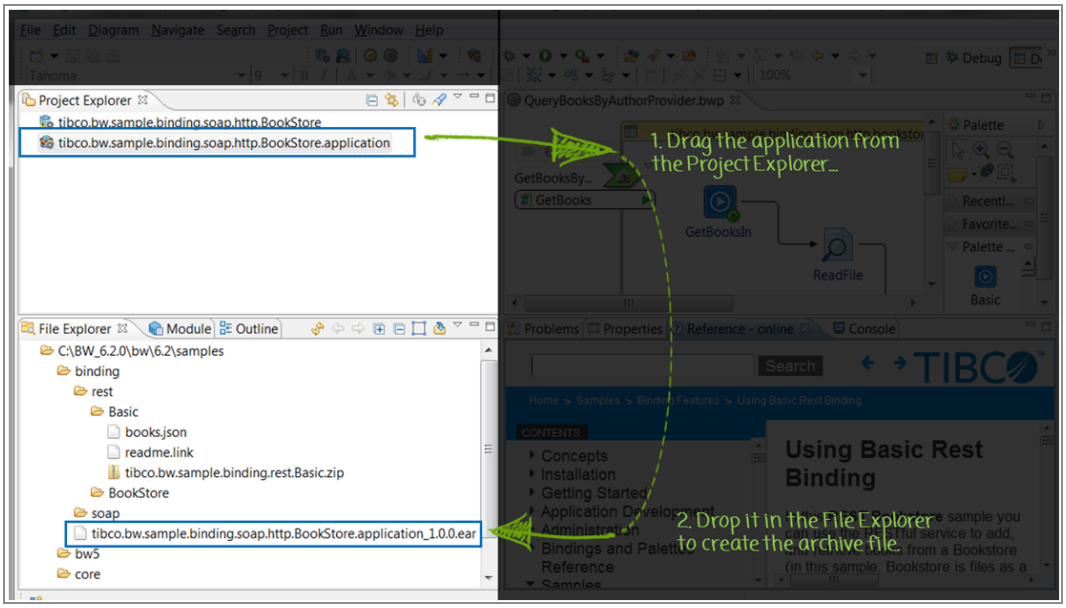
Creating an Application Archive

You can create an application archive in TIBCO Business Studio by dragging and dropping the application project from the Project Explorer to the File Explorer window.

You create an application archive after designing and testing the application.

Procedure

1. In TIBCO Business Studio, go to **File Explorer** tab and click the **Open Directory to Browse** icon.
2. Select the directory where you want to store the archive and click **OK**.
3. Drag an application from the Project Explorer to the directory in the File Explorer.



The application archive is written to the directory using the syntax `<application>_<version>.ear` where the `<version>` starts from 1.0.0 and increments as more generations occur.

Uploading an Application Archive

Uploading an application archive copies the specified file to the specified domain. The archive is copied to the *BW_HOME*\domains*domain_name*\archives directory. If the specified archive exists in the domain, it can be replaced on upload.

The following procedure shows how to upload an application archive.

BWAdmin Command Line

Procedure

1. Run the following command from the command line. Specify the fully qualified location of the application archive file. Note the use of forward slashes "/" for the Windows path. This Windows example uploads the BookStore sample that has been copied to the `C:\ear` folder.

   ```
   BW_HOME\bin>bwadmin upload -d MyDomain
   C:/ear/tibco.bw.sample.binding.rest.BookStore.application_1.0.0.ear
   ```

   Tip: If the specified application archive exists in the selected domain, use the `-replace` option to replace it.

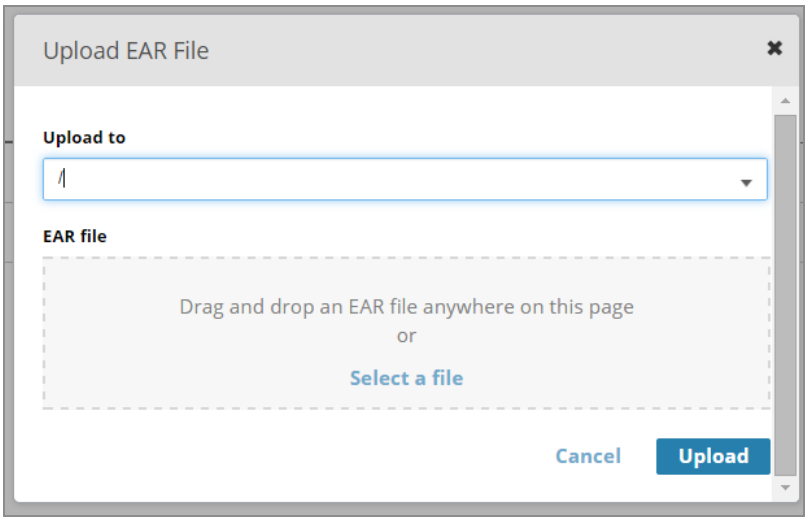
2. By default, the `upload` command copies the archive file to the *BW_HOME*\domains*domain_name*\archives folder. To upload an archive to a different location, use the `-path` option and specify a path relative to the *BW_HOME*\domains*domain_name* folder. For example, the following example creates the *BW_HOME*\domains*domain_name*\test folder and uploads the archive file:

   ```
   BW_HOME\bin>bwadmin upload -d MyDomain -path ../../test/MyArchives
   C:/ear/tibco.bw.sample.binding.rest.BookStore.application_1.0.0.ear
   ```

Admin UI

Procedure

1. Select the domain and open the **Application Archives** page. Click **Upload**.
2. In the **Upload EAR File** dialog, enter the following information:
   - **Upload to**: EAR file upload folder. Default location is the `domains` folder.
   - **EAR file**: Drag and drop the archive file. If the file exists in that folder, select the **Replace any existing version** checkbox.



3. Click **Upload**, then **Done**.

   The archive is displayed on the Application Archives page.

Configuring Application Archives

An application archive containing encrypted profiles require keystore configuration.

To use a non-encrypted profile when deploying an application archive, there is no need to configure an application archive.

This functionality is not supported with the local mode.

BWAdmin Command Line

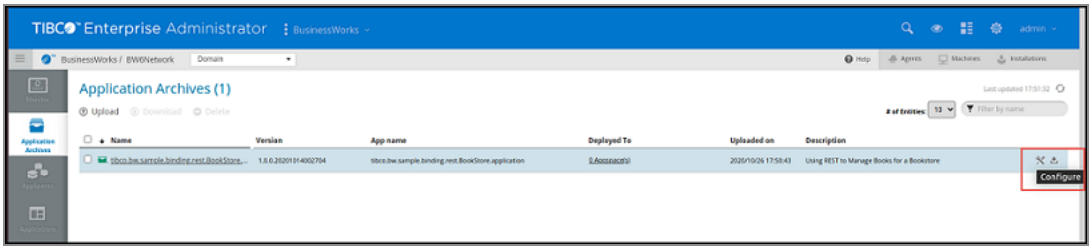
Use the following command line to configure an application archive:

```
BW_HOME\bin>bwadmin config -d <domain name> -keyStoreFile <keystore file
path> -keyAlias <alias name> -keyStorePassword <keystore password> -
keyStore <keystore file name> -keyAliasPassword <key alias password>
archive <name of archive>
```

Admin UI

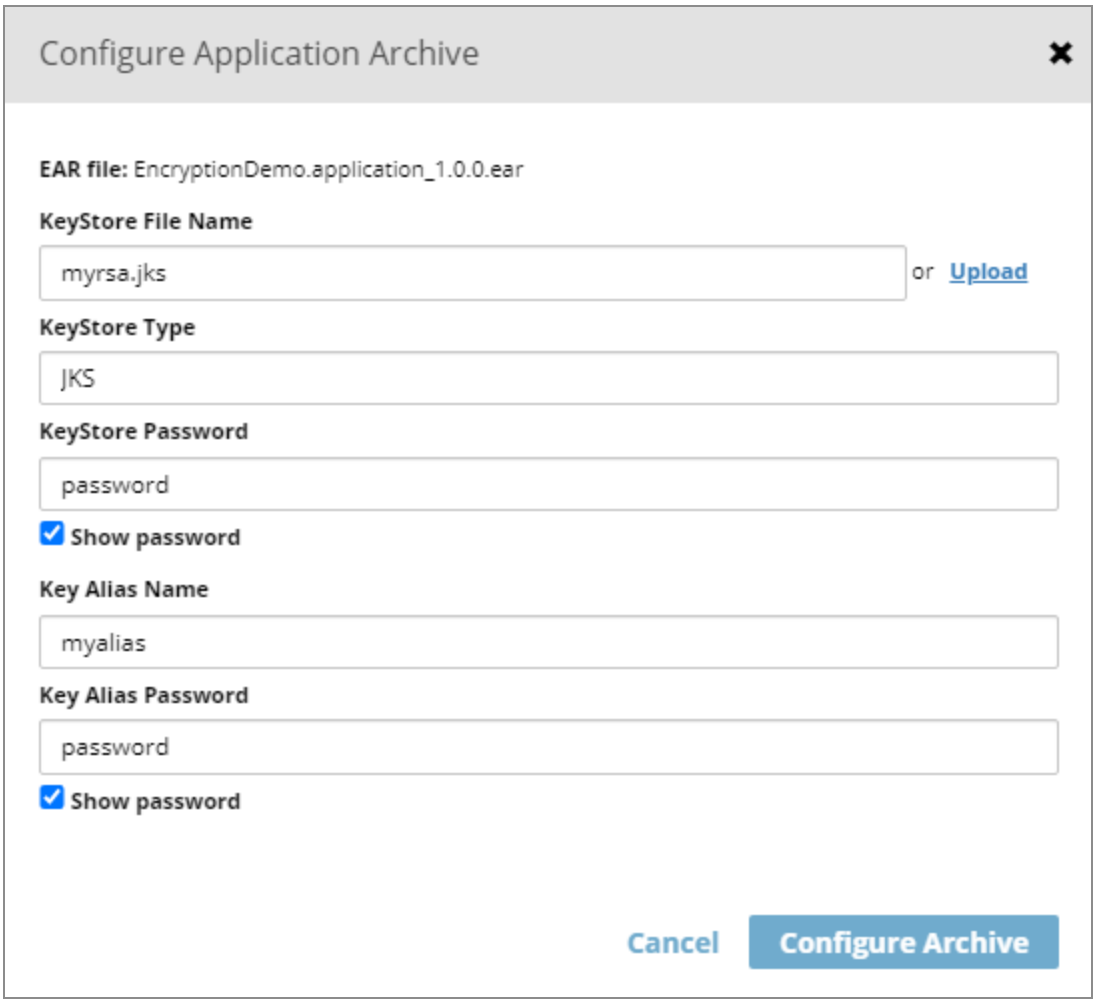
Procedure

1. Go to Application Archive Level 1 page and click **Configure**.



2. Optionally, go to the Application Archive Level 2 page and click **Configure**.

   You can upload the Keystore file to the database by clicking the **Upload** button.

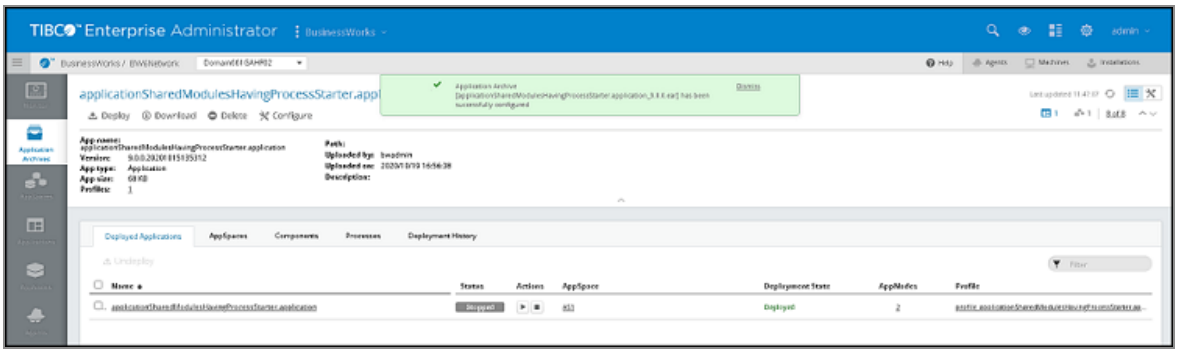


3. Provide the following encryption settings:
    - Keystore File Name
    - Keystore Type - It is automatically populated based on the type of keystore file provided. The following keystore file types are supported:
        - JKS
        - JCEKS
        - PKCS12
    - Keystore Password
    - Key Alias Name

- Key Alias Password

4. Click **Configure Archive**.

   A success message is displayed.



> **Note:** After configuring the archive, a copy of the Keystore file is stored in the database. When the domain is spanned, the Keystore copy file is created in the remote agent.

REST API

To perform the same operation from REST API, follow the below APIs to encrypt, decrypt, and configure the archive:

Encrypt profile:

```
http://<Hostaddress>:<port>/bw/v1/domains/<domain_
name>/archives/encrypt?name=<archive_name>
```

Decrypt profile:

```
http://<Hostaddress>:<port>/bw/v1/domains/<domain_
name>/archives/decrypt?name=<archive_name>
```

Configure the archive:

```
http://<Hostaddress>:<port>/bw/v1/domains/<domain_
name>/archives/config?name=<archive_name>
```

For more information, see Accessing the BWAgent REST API with the Swagger UI.

Deploying an Application

You use the `deploy` command to deploy an application archive to an AppSpace.

You can deploy multiple applications to an AppSpace. You can deploy and run multiple versions of the same application on the same AppNode at the same time.

BWAdmin Command Line

When using the `deploy` command, the EAR filename is the relative location of the archive with respect to the *BW_HOME*`\domains\`*domain_name*`\archives` folder. For example, if `MyDomain` contains the AppSpace to deploy to and the archive file is in the *BW_HOME*`\domains\MyDomain\archives` directory, do not specify a qualifier for the archive file location.

Run the following command from the command line to deploy an uploaded application archive to `MyAppSpace`:

```
BW_HOME\bin>bwadmin deploy -d MyDomain -a MyAppSpace
tibco.bw.sample.binding.rest.BookStore.application_1.0.0.ear
```

Replacing a deployed archive file undeploys the application. To replace an archive file that has been uploaded, but not deployed, use the `-replace` option with the upload command to upload the archive again. When the `-replace` option is used to replace an archive file that has been uploaded and deployed, the following error message is displayed,

```
Upload Operation Cancelled. As Application MyApplication from archive
MyApplication.ear has been deployed to those AppSpaces.
Please use force replace (-f)option with the upload command i.e
[upload -domain D -replace -f C:/Users/Administrator/Desktop/EAR/MyApplication.ear]
```

The force replace option undeploys the existing archive file and replace the old archive file with the new file.

Admin UI

Procedure

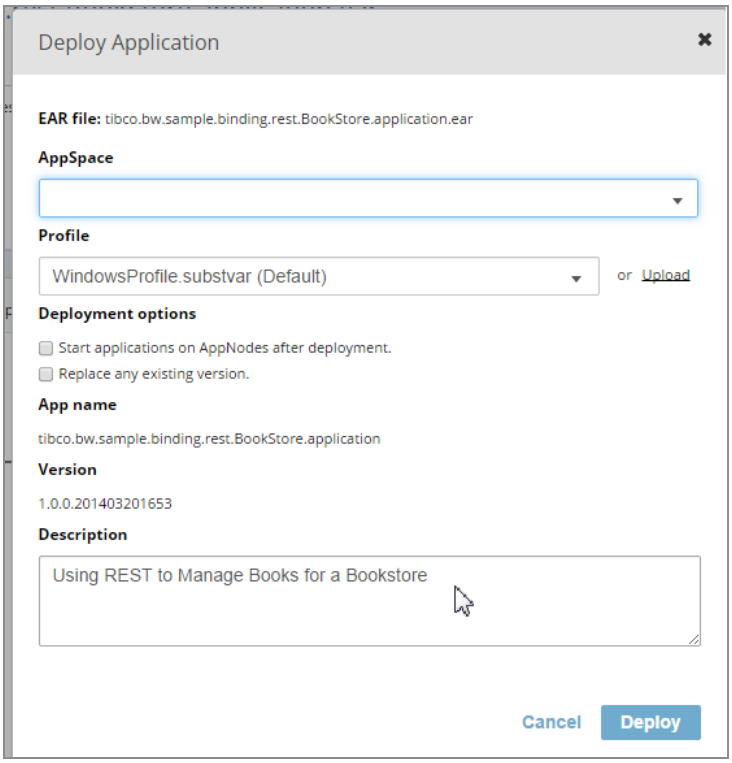
1. Select the domain and open the **Application Archives** page. Drill down to an archive. On the **Application Archive** page for the selected archive, click **Deploy**.
2. In the **Deploy Application** dialog, enter the following information:
   - **AppSpace**: The AppSpace to deploy to.
   - **Profile**: The profile file to use for deployment. (An application profile can be

changed after deployment. See Configuring an Application. Click **Upload** to upload a new profile.

- **Start applications on AppNodes after deployment**: Starts the application on AppNodes after deployment.
- **Replace any existing version**: Replaces an existing version of the same application. If the **Replace any existing version** checkbox is selected, a warning message is displayed, indicating that the upload action undeploys the existing application from the AppSpace.

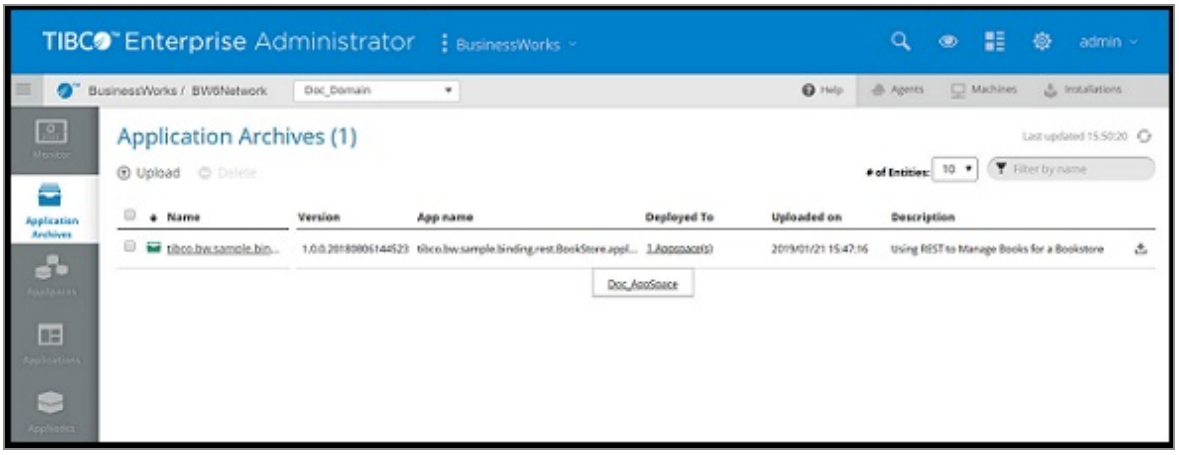
> **Note:** If the EAR file is uploaded but not deployed, the warning message is not displayed and the EAR file is replaced.

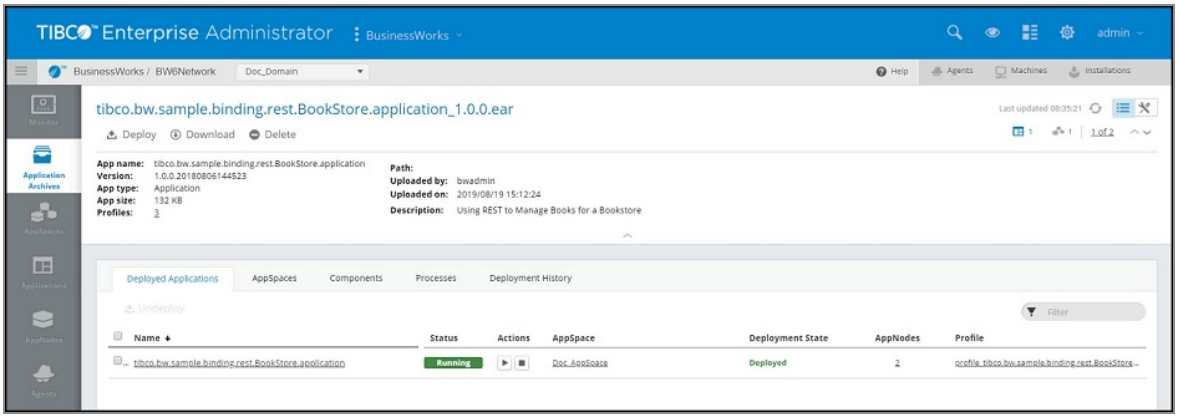
- **App name/Version:** Read-only fields that display the application name and version.
- **Description**: Optional description; pulled from archive.



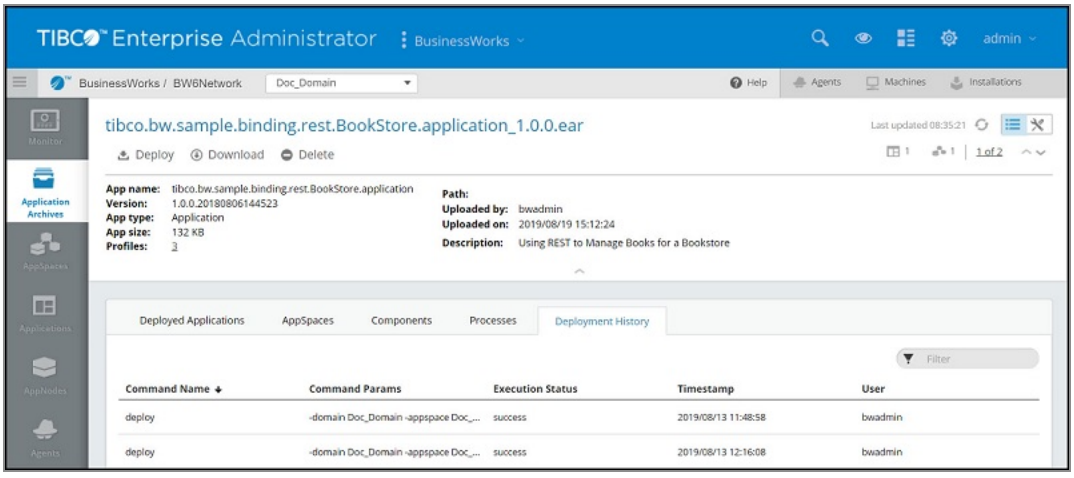
The archived applications are displayed on the **Application Archives** page. The **Deployed To** column displays the number of AppSpaces where the application has

been deployed.

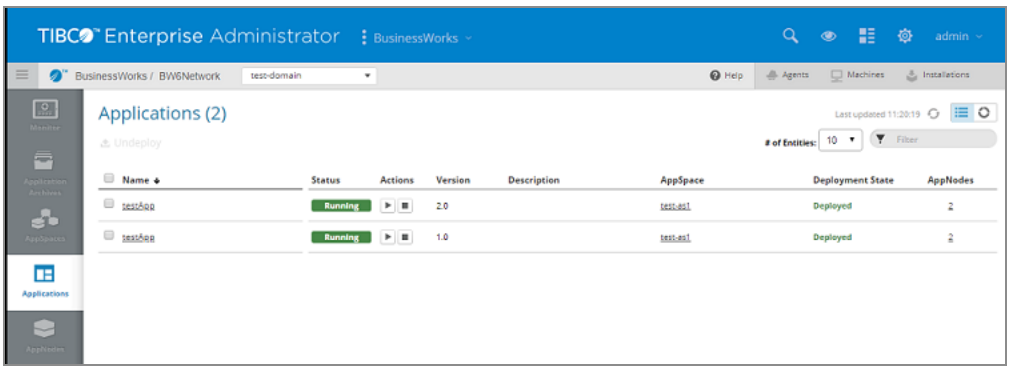


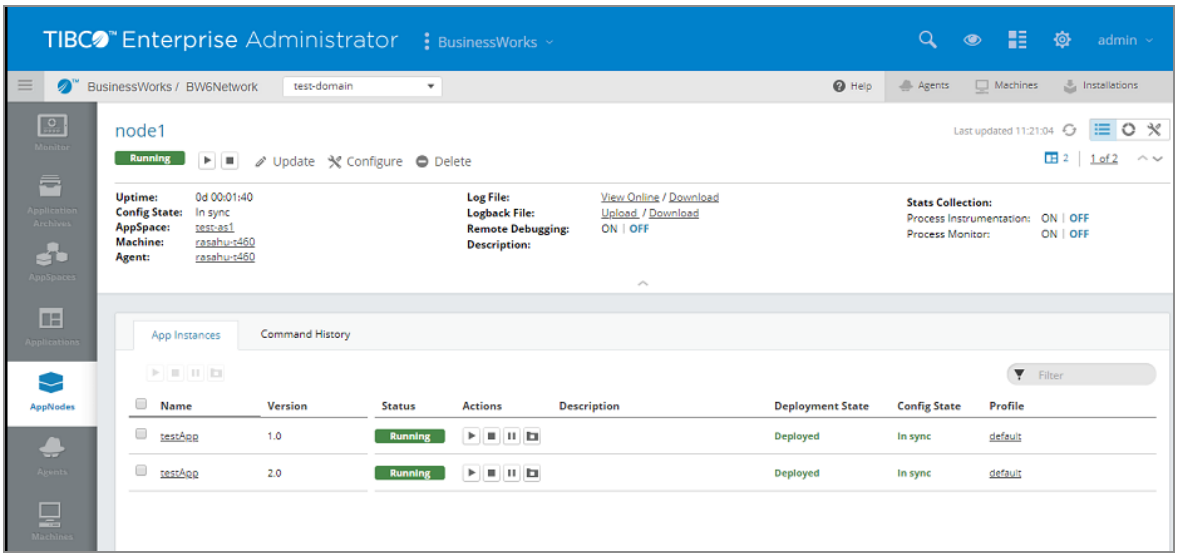


To view deployment history of the application archive, open the **Deployment History** tab. You can view the commands that were issued on an application archive, the execution status of the commands and the timestamp.



To view multiple versions of the same application, open the **Applications** page.





Downloading an Application Archive

You can download an application archive on your local system. The following steps show how to download an application archive.

BWAdmin Command Line

Procedure

1. Run the following command from the command line to download all application archive files in a specified domain. Note the use of forward slashes "/" for the Windows path.

```
BW_HOME\bin>bwadmin download -d MyDomain -s C:/Tmp/Archives/
```

2. To download a specific application archive file:

```
BW_HOME\bin>bwadmin download -d MyDomain -s C:/Temp/Archives -a
Application_Name.ear
```

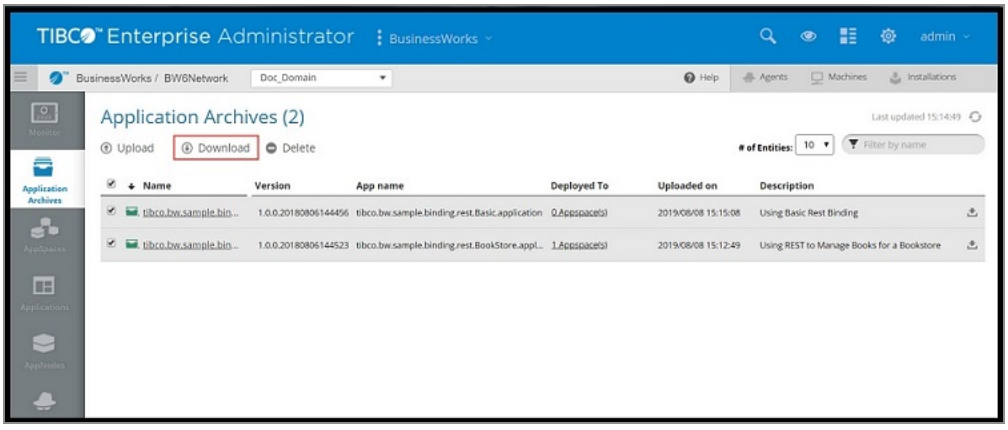
Note: For more information about all supported options, use the following command:

```
BW_HOME\bin>bwadmin download --help
```

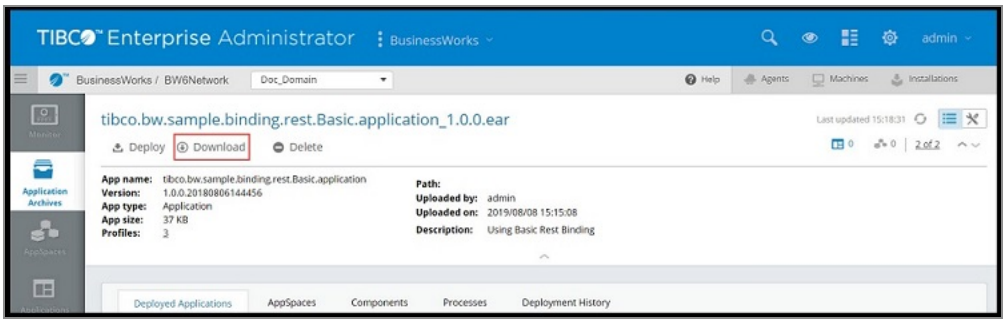
Admin UI

Procedure

1. Select the domain and open the **Application Archives** page.
2. Select one or more application archives and select **Download**.



Select any application archive and download it from Page 2.



Editing Application and Application Instance Properties

Application or application instance properties can be changed after deployment. You can then export a configuration and apply it to another application or application instance.

BWAdmin Command Line

The profile for an application is located under the `META-INF` folder in the application's archive. The profile is the file with the extension, `.substvar`. For more information about generating more than the default profile for an application, see Creating an Application with Multiple Profiles.

When the `config` command is applied to an application, the profile changes are applied to all application instances by default. To apply a profile change to a specific application instance, use the `config -appnode` option to identify the specific AppNode.

If the archive contains the `WindowsProfile.substvar` file, use the following command to update the profile:

```
BW_HOME\bin>bwadmin config -d MyDomain -a MyAppSpace
-n MyAppNode -p WindowsProfile.substvar application
MyApplication
```

If you want to use encrypted profiles, make sure to configure your application archive. For more details, see Configuring an Application Archive.

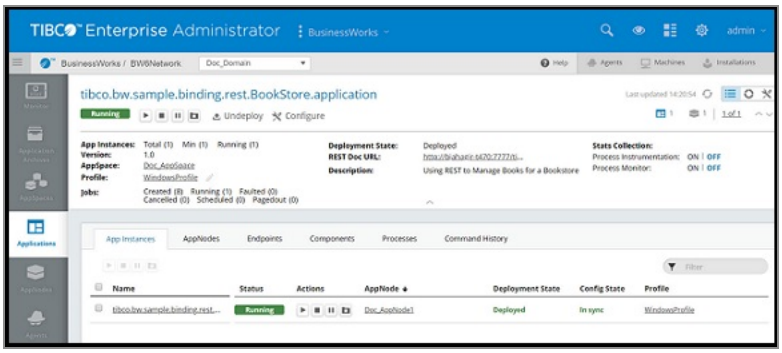
Admin UI

From the Admin UI you can change the profile for all instances of the application, or for a single application instance.

Procedure

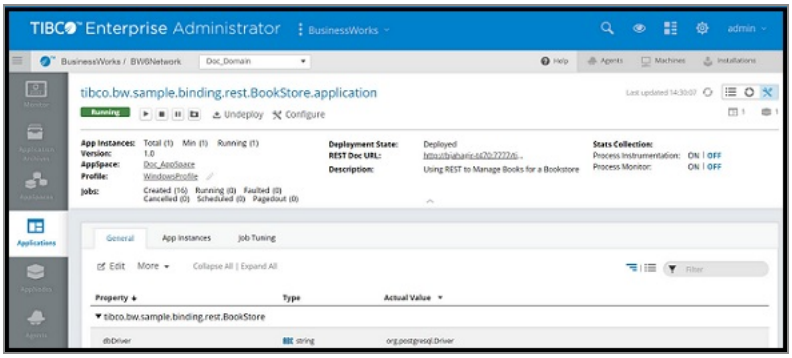
1. Select the application that you want to configure on the **Applications** page.

   The **Application** page is opened, where you can view the application instances, the AppNodes each instance is deployed to, the deployment state, and the applied profile file.



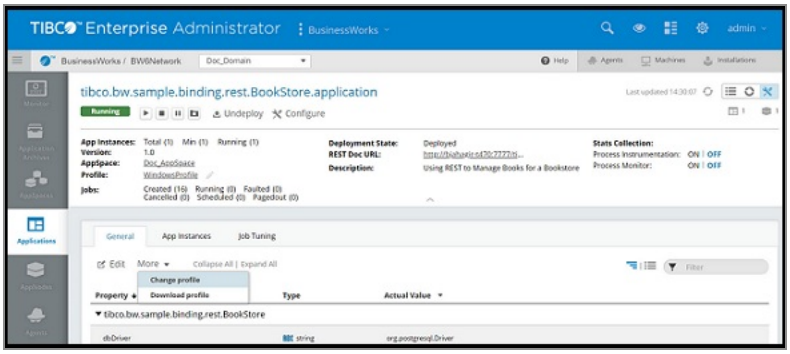
2. To edit application or application instance properties, click the **Edit** icon in the upper right of the **Applications** page.

   The **Application Properties** page is displayed.



3. Use the **General** tab to edit application properties and the **AppInstances** tab to edit application instance properties.
4. Click **Edit** on the **General** tab to edit application properties.
   a. Click **Submit** to save the property changes and apply them to the application.

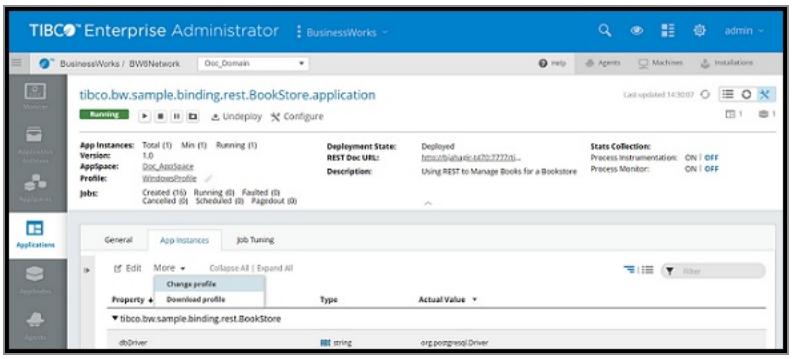
b. Click **More > Change Profile** to open the **Change Profile** dialog where you can select a new profile to apply to the application or upload a new profile. Restart the application to apply the new profile.



5. Click **Edit** on the **AppInstances** tab to edit application instances properties. Select the instance (by AppNode) you want to edit.

   a. Click **Submit** to save the property changes and apply them to the application instance.

   b. Click **More > Change Profile** to open the **Change Profile** dialog where you can select a new profile to apply to the application instance or upload a new profile. Restart the application to apply the new profile.



Exporting an Application Profile

An application profile can be exported from the application archive with the `export` command or from the Admin UI. After an application is configured with a profile, it becomes part of the application archive. An application configuration can be used to configure another application. If property changes are required after deployment, export the profile, edit, and deploy with the edited profile file.

BWAdmin Command Line

Configurations are exported to the file system in the working directory.

The configuration is saved to: *application_name_profile_name*.substvar

To export a profile, run the following command at the command line:

```
BW_HOME\bin>bwadmin export -d MyDomain -a MyAppSpace application
tibco.bw.sample.binding.rest.BookStore.application 1.0
```

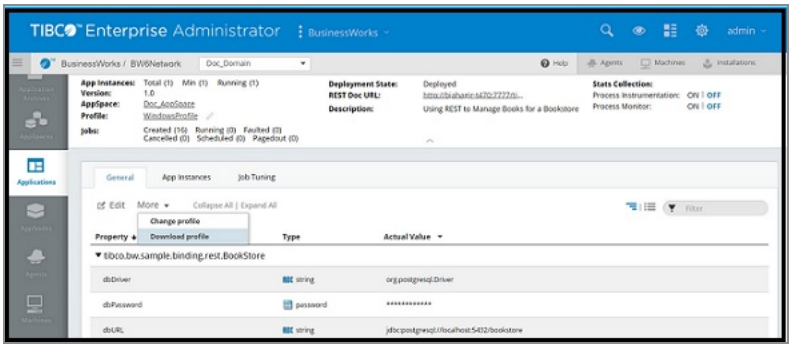
The application configuration file `tibco.bw.sample.binding.rest.BookStore.application_WindowsProfile.substvar` is written to the working directory.

Admin UI

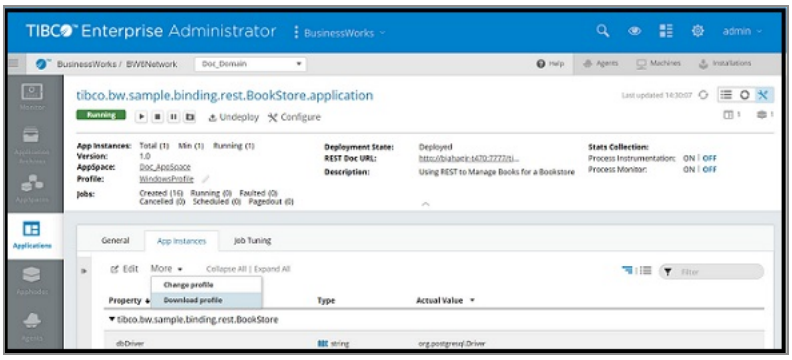
From the Admin UI you can export the profile for an application, or for a single application instance.

Procedure

1. Open the **Application** page and drill down into the application.
2. Click the **Edit** icon in the upper right of the page. The **Application Properties** page is displayed. The **General** tab displays application properties and the **AppInstances** tab displays application instance properties.
3. Click **More > Download Profile** on the **General** tab to export the application profile.



4. Click **More > Download Profile** on the **AppInstances** tab to export the application instance profile.



Starting an Application

To start an application after deployment, run the `start` command or click the **Start** icon in the Admin UI. If you stop an AppNode for a running application, the application state is persisted when you restart the AppNode.

BWAdmin Command Line

When an application archive is deployed, the default action starts the application on each AppNode defined in the AppSpace. However, an archive file can be deployed with the `–startondeploy` option set to *false* so it is not started after deployment. Then, the `start` command can be used with the `–appnode` option to start the application on a specific AppNode.

Procedure

1. Start the AppSpace.
2. Run the `start` command for the application. For example:

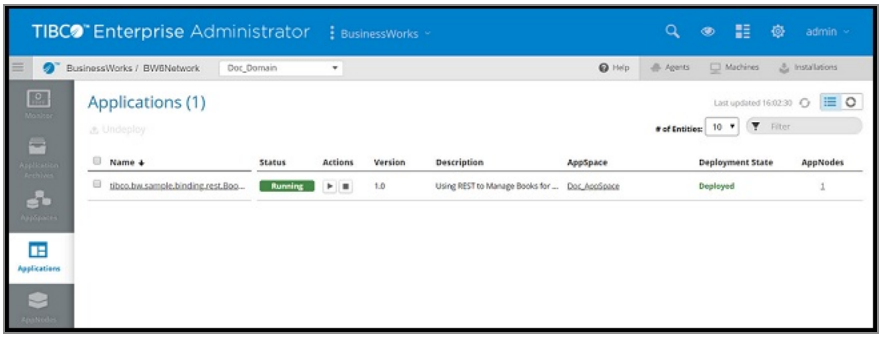
```
BW_HOME\bin>bwadmin start -d MyDomain -a MyAppSpace application
tibco.bw.sample.binding.rest.BookStore.application 1.0
```

   To start any specific application instance, use the following command:

```
BW_HOME\bin>bwadmin start -d MyDomain -a MyAppSpace -n MyAppNode
application tibco.bw.sample.binding.rest.BookStore.application 1.0
```

Admin UI

To start the application, click the **Start** icon on the **Applications** page. The AppSpace and AppNodes must be running.



Viewing Running Applications

Use the BWAdmin `show` command to verify a running application, or view the application in the Admin UI.

BWAdmin Command Line

Run the `show` command to see application and configuration status.

```
BW_HOME\bin>bwadmin show -domain MyDomain -appspace MyAppSpace
applications
```

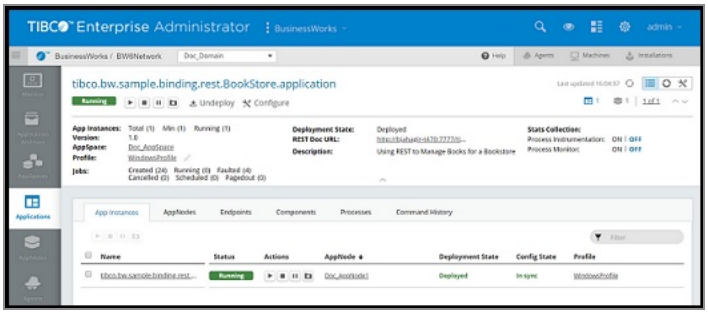
To view status of any specific application instance, use the following command:

```
BW_HOME\bin>bwadmin show -domain MyDomain -appspace MyAppSpace -n
MyAppNode applications
```

Admin UI

View the application's status on the **Application** page. (To open this page, drill into the application from the **Applications** page.) The Admin UI displays the following information:

- Total number of application instances, the minimum number of instances (AppNodes), and running number of instances.
- Application version.
- AppSpace
- Number of created jobs, running jobs, and faulted jobs.
- The applied profile.
- The deployment state.
- The REST Doc URL for applications using REST services. Click the link to open the REST UI page where you can test out operations. (The application must be running.)
- The application description.



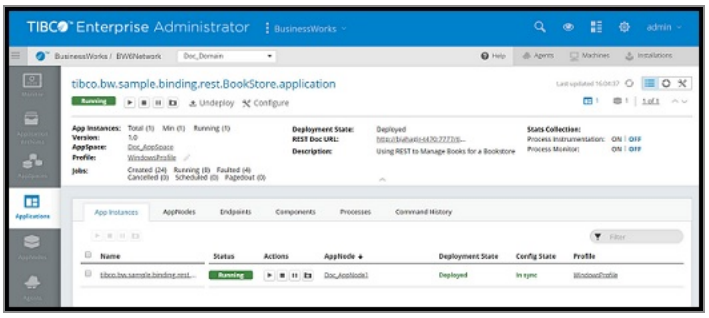
Viewing Endpoints, Components, Processes, and Command History

You can view endpoints, components, processes and the command history for a running application from the Admin UI.

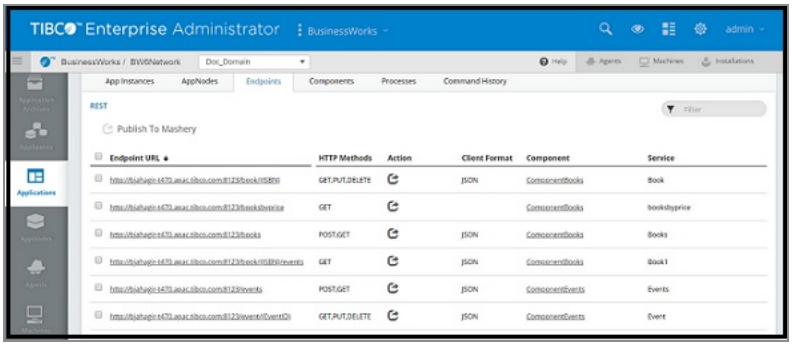
Admin UI

Procedure

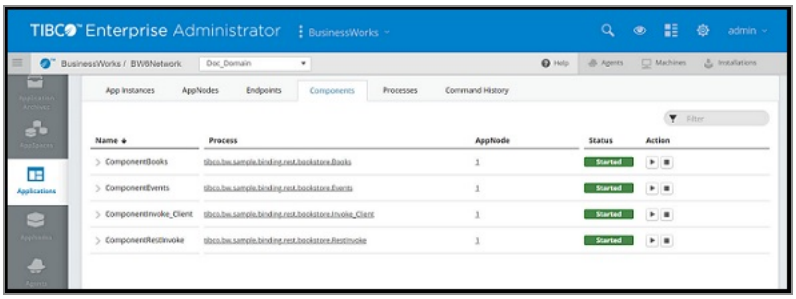
1. On the **Applications** page, select the running application you want to view details to configure.



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



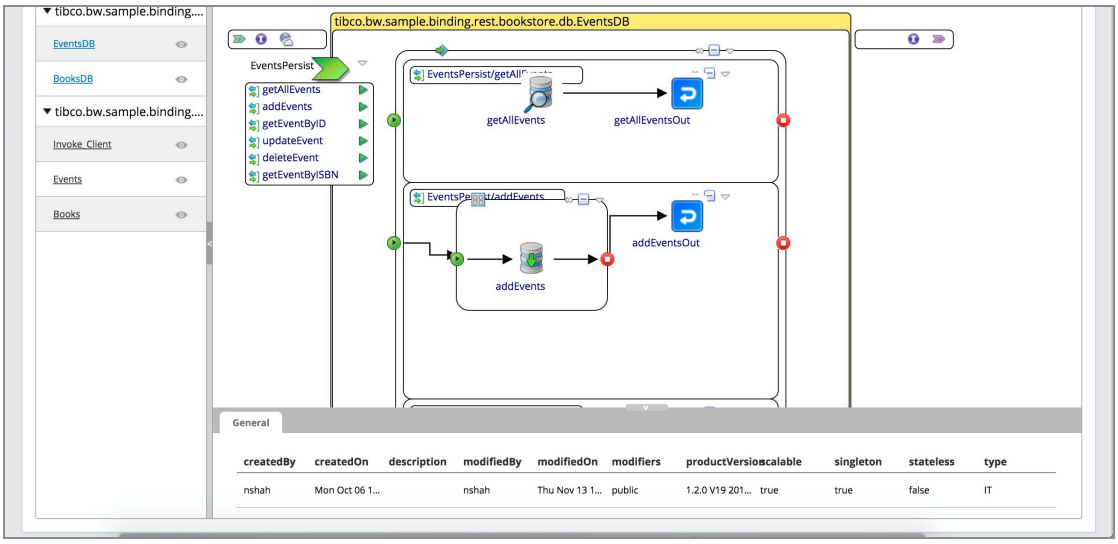
3. Open the **Components** tab to view components in the application.



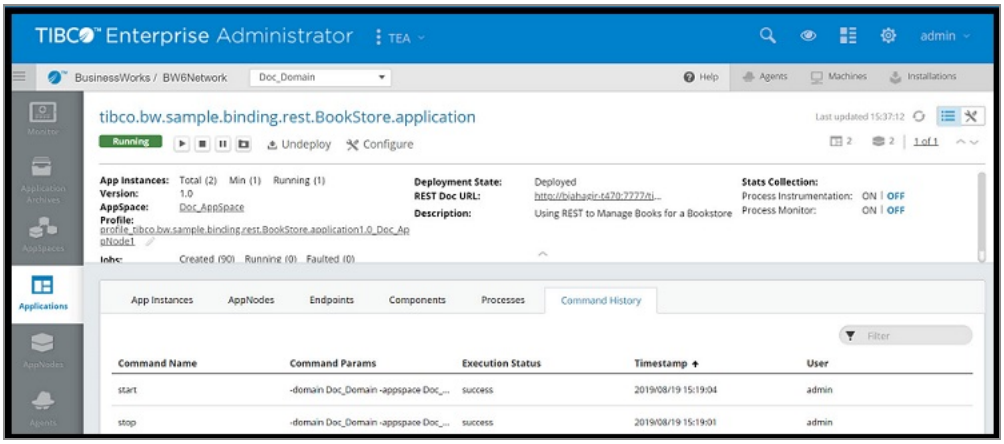
4. Open the **Processes** tab to view application processes.

   If the application archive file was generated using ActiveMatrix BusinessWorks 6.2.x or higher, you can view expand a process and click it to view the SVG process diagram.

> **Note:** When viewing process diagrams through the Admin UI, the optimal resolution is 1920 x 1080 pixels or higher.



5. Open the **Command History** tab to view the commands or operations that were performed on an Application.



Configuring a Unified Doc URL

You can configure a unified documentation URL for all applications that use REST services that are running in a single AppSpace. Alternatively, the documentation URL can be configured for each AppNode in an AppSpace.

Documentation endpoint configuration properties can be specified at the AppSpace or the AppNode level. The details specified at the AppSpace level apply to all applications running on all AppNodes within the AppSpace. Properties set at the AppNode level only apply to applications running on that AppNode.

Properties are configured in the AppSpace or AppNode configuration file or configuration template file.

Procedure

1. To set documentation endpoint properties at the AppSpace level:
    a. Copy the existing AppSpace configuration template file `appspace_config.ini_template` (in *`BW_HOME`*`/config/`) to a temporary location.
    b. Uncomment and configure the following properties in the **BW REST Swagger Configuration** section of the file:

```
# ------------------------------------------------------------
------------------
# Section:  BW REST Swagger Configuration.  The properties in
this section
# are applicable to the Swagger framework that is utilized by
the BW REST
# Binding.
#
# Note: There are additional BW REST Swagger configuration
properties that
# can be specified in the BW AppNode configuration file
"config.ini".  Refer to
# the BW AppNode configuration file's section "BW REST Swagger
configuration"
# for details.
# ------------------------------------------------------------
------------------
# Swagger framework reverse proxy host name.  This property is
optional and
# it specifies the reverse proxy host name on which Swagger
framework serves
# the APIs, documentation  endpoint, api-docs, and so on.
#bw.rest.docApi.reverseProxy.hostName=localhost

# Swagger framework port.  This property is optional and it
specifies the
# reverse proxy port on which Swagger framework serves the
APIs, documentation
# endpoint, api-docs, and so on.
#bw.rest.docApi.reverseProxy.port=0000
```

2. To set documentation endpoint properties at the AppNode level:
    a. Copy the existing AppNode `config.ini` file (in the root of the AppNode folder),

or the `appnode_config.ini_template` (in *`BW_HOME`*`/config/`) file, to a temporary location.

b. Configure the following properties in the **BW REST Swagger Configuration** section of the file (note that the port property is uncommented by default):

```
# ------------------------------------------------------------
-----------------
# Section:  BW REST Swagger configuration.  The properties in
this section
# are applicable to the Swagger framework that is utilized by
the BW REST
# Binding.
#
# Note: There are additional BW REST Swagger configuration
properties that
# can be specified in the BW AppSpace configuration file
"config.ini".
# Refer to the BW AppSpace configuration file's section
# "BW REST Swagger configuration" for details.
# ------------------------------------------------------------
-----------------
# Swagger framework host name.  This property is optional and
it specifies the
# host name on which Swagger framework serves the APIs,
documentation endpoint,
# api-docs, and so on.  The default value is the host name on
which the BW AppNode
# is run.
#bw.rest.docApi.hostName=localhost

# Swagger framework port.  This property is required and it
specifies the port
# on which Swagger framework serves the APIs, documentation
endpoint,
# api-docs, and so on.
bw.rest.docApi.port=7777
```

3. Use one of the following `config` admin commands to push the configuration to the AppSpace or the AppNode:
   - AppSpace:

```
bwadmin[admin] > config -d myDomain -a myAppSpace -cf
```

```
                                        <temporaryLocation>/config.ini
```

- AppNode:

```
bwadmin[admin]> config -d myDomain -a myAppSpace -n myAppNode -
cf <temporaryLocation>/config.ini
```

Result

Documentation for all applications in the AppSpace that use REST services is available at the given URL. You can open the documentation URL by clicking the REST Doc URL link for the running application in the Admin UI or by opening the URL at the specified host name and port. The application must contain REST services and must be running. If you configured the documentation URL just for the AppNode, the documentation for applications running in the specified AppSpace is available at the given URL.

Stopping an Application

To stop a running application after deployment, run the `stop` command or click the **Stop** icon in the Admin UI. Applications or application instances can be stopped. Stop an application before undeploying.

BWAdmin Command Line

Run the following command at the command line to stop an application:

```
BW_HOME\bin>bwadmin stop -d MyDomain -a MyAppSpace application
tibco.bw.sample.binding.rest.BookStore.application 1.0
```

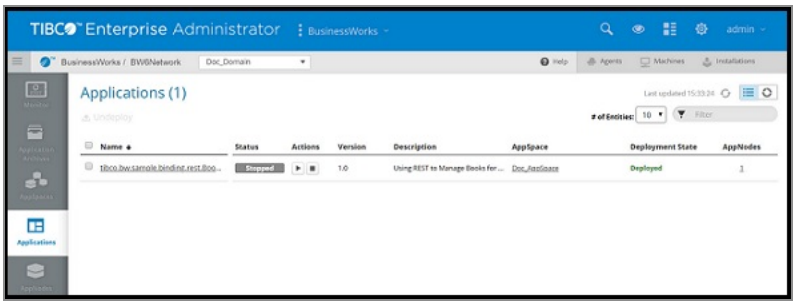
To stop an application instance, use the following command:

```
BW_HOME\bin>bwadmin stop -d MyDomain -a MyAppSpace -n MyAppNode
application tibco.bw.sample.binding.rest.BookStore.application 1.0
```

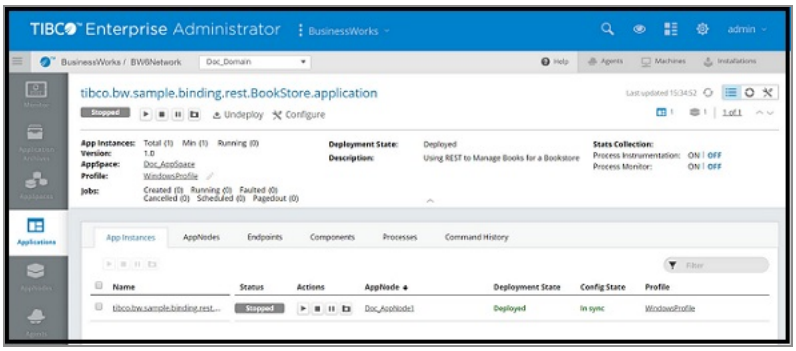
Admin UI

Procedure

1. To stop an application, click the **Stop** icon for the application on the **Applications** page. The status of the application instance changes from Running to Stopping, a transient state, then Stopped:



2. To stop an application instance, click the **Stop** icon for the instance on the **App Instances** tab. The status of the application instance changes from Running to Stopping, a transient state, then Stopped:



Undeploying an Application

Undeploying an application removes the deployed application from the AppSpace.

BWAdmin Command Line

Run the following command to undeploy:

```
BW_HOME\bin>bwadmin undeploy -d MyDomain -a MyAppSpace application
tibco.bw.sample.binding.rest.BookStore.application 1.0
```

Admin UI

Procedure

1. On the **Applications** page, select the application checkbox and click **Undeploy**.
2. The **Undeploy Application(s)** dialog is displayed. To undeploy all process monitoring historical data from the database, select the **Remove Process Monitoring related historical data of selected apps from DB** checkbox. Click the **Undeploy** button to undeploy the application.

Starting a Component in an Application

BWAdmin Command Line

To start a component in an application, run the `startcomponent` command or click the **Stop** icon in the Admin UI.

When an application is started, the default action starts all the components of the application on each AppNode, defined in the AppSpace. However, a component can be stopped using the `stopcomponent` command. The `startcomponent` command can then be used with the `-appnode` option to start a component of an application on a specific AppNode.

Procedure

1. Start the AppSpace.
2. Run the `startcomponent` command for the component of an application. For example:

```
BW_HOME\bin>bwadmin startcomponent -d MyDomain -a MyAppSpace -n
MyAppNode MycomponentName tibco.bw.sample.MyApp.application 1.0
```

Note: To auto start a component with process starter activity, use `enablecomponentautostart` command. For example:

```
BW_HOME\bin>bwadmin enablecomponentautostart -d MyDomain -a
MyAppSpace -n MyAppNode ComponentReceiver
jmsSenderReceiver.application 1.0
```

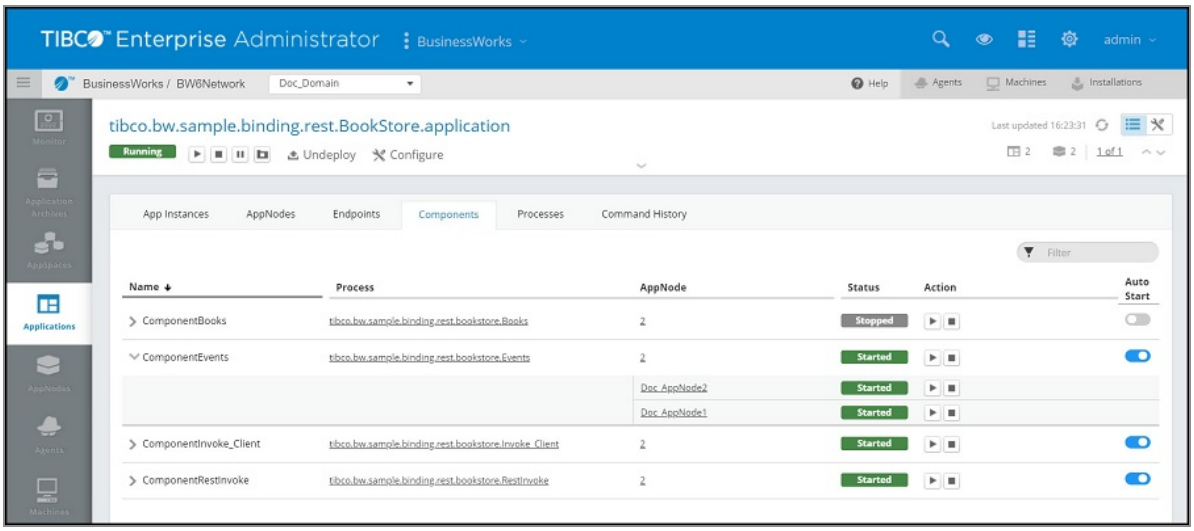
This functionality is applicable at AppSpace level and not at AppNode level.

Admin UI

When an application is started, the default action starts all the components of the application on each AppNode defined in the AppSpace. Click the **Components** tab to view the components in the application. Click the **Start** icon to start all the components in the AppNodes. If all the components are running, the **Status** changes to started. The status bar displays the number of components running on the AppNodes. On hovering over the status bar, you can see the number of running components. To start a component on a specific AppNode, click the **right angle bracket** icon on the left to view the list of AppNodes the component is running on. Click the **Start** icon against the component you want to start.

Note: The **right angle bracket** icon, which collapses the list of AppNodes that the component is running on, is enabled only after the application has started.

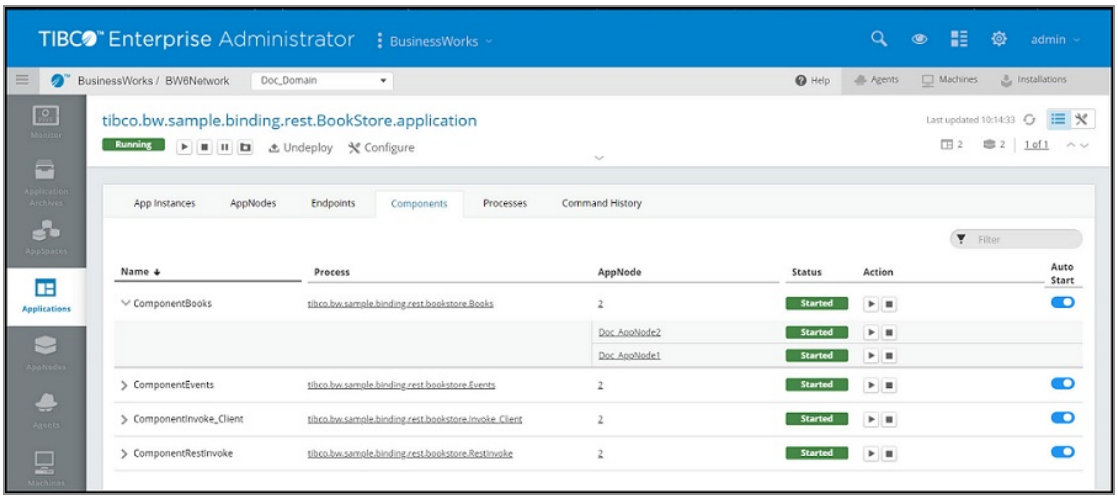
Note: Starting the top level **Start** icon starts all the components with auto start toggle button on for all the AppNodes.



To enable auto start of a component, perform the following steps:

Procedure

1. Select Application Level 2 page and select the **Component** tab.
2. Use the toggle button in the **Auto Start** column for each component process to decide whether to enable auto start of a component process or not during execution.



3. Restart an application for the changes to take effect.

Stopping a Component in an Application

To stop a component of an application, run the `stopcomponent` command or click **Stop** icon in the Admin UI.

BWAdmin Command Line

When an application is started, the default action starts all the components of the application on each AppNode, defined in the AppSpace. However, the `stopcomponent` command can be used with the `-appnode` option to stop a component of the application on a specific AppNode.

Procedure

1. Start the AppSpace.
2. Run the `stopcomponent` command for the component of an application. For example:

```
BW_HOME\bin>bwadmin stopcomponent -d MyDomain -a MyAppSpace -n
MyAppNode MycomponentName tibco.bw.sample.MyApp.application 1.0
```

Note: To disable auto start with process starter activity, use the `disablecomponentautostart` command. For example:

```
BW_HOME\bin>bwadmin disablecomponentautostart -d D -a AS -n
AN ComponentReceiver jmsSenderReceiver.application 1.0
```

This functionality is applicable at AppSpace level and not at AppNode level.

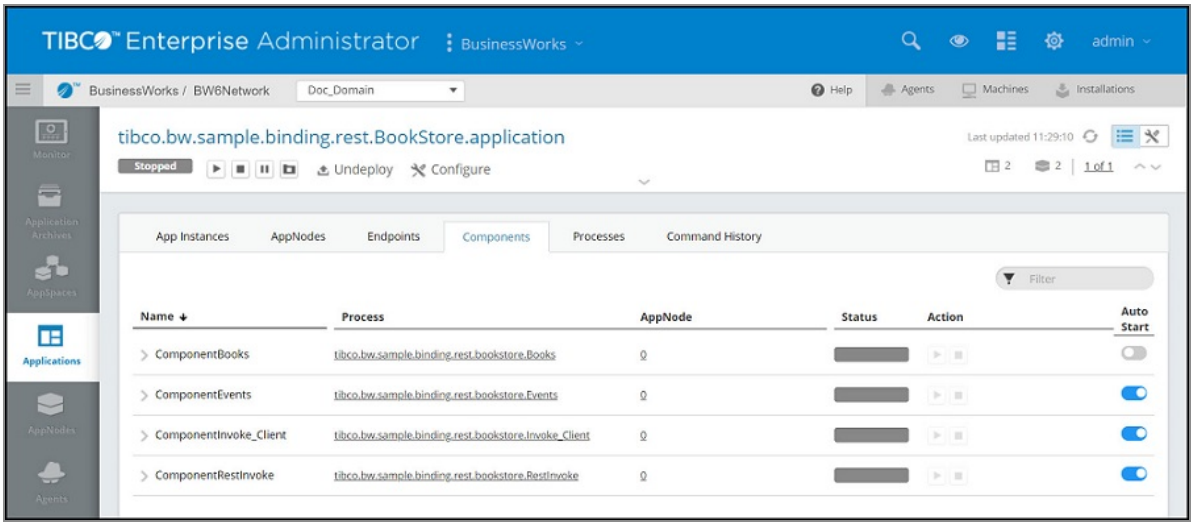
Admin UI

When an application is started, the default action starts all the components of an application on each AppNode defined in the AppSpace. Click the **Components** tab to view the components in the application, and select the running application you want to stop the components for. To stop a component on a specific AppNode, click the icon on the left to view the list of AppNodes the component is running on, and click the **Stop** icon against the component you want to stop. The status bar displays the number of components that have stopped running on the AppNodes. On hovering over the status bar,

in the primary row table, you can see the number of components that are running and the number of components that have stopped running on the AppNode.

Note: The collapse icon is enabled only after an application has started.

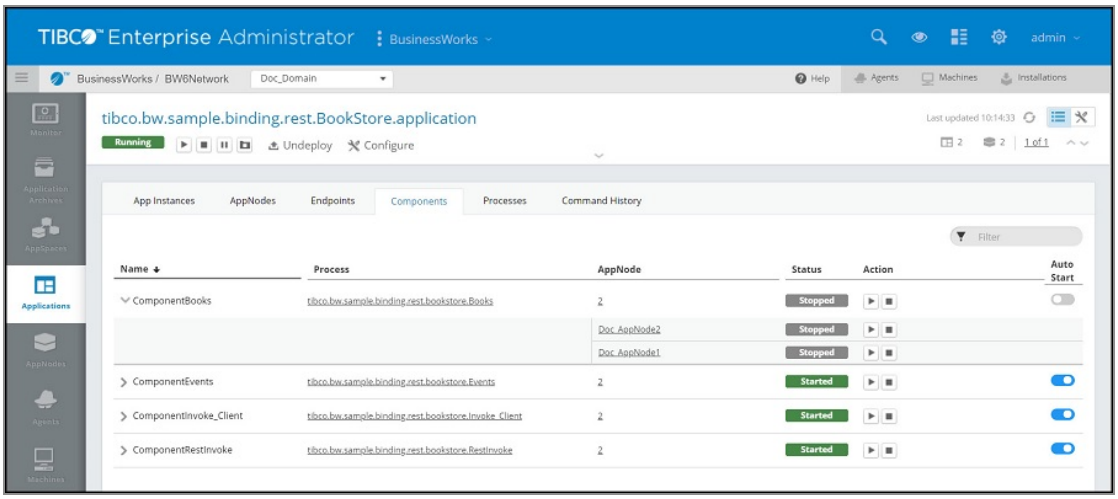
Note: Stopping the top level **Stop** icon stops all the running components on all the AppNodes.



To disable auto start of a component:

Procedure

1. Select Application Level 2 page and select the **Component** tab.
2. Use the toggle button in the **Auto Start** column for each component process to decide whether to enable auto start of a component process or not during execution.



3. Restart an application for the changes to take effect.

Retrieving List of Components in an Application

To get the list of components of an application, run the `getcomponents` command or click the **Components** tab in the Admin UI.

BWAdmin Command Line

Procedure

1. Start the AppSpace.
2. Run the `getcomponents` command for an application. For example:

```
BW_HOME\bin>bwadmin getcomponents -d MyDomain -a MyAppSpace -n
MyAppNode tibco.bw.sample.MyApp.application 1.0
```

Admin UI

On the **Applications** page, select the running application you want to view the components for. Open the **Components** tab to view the components in the application.

Retrieving Details of a Component in an Application

To get the details of a particular component of an application, run the `getcomponentdetail` command or view the details under the **Status** label, in the Admin UI.

BWAdmin Command Line

Procedure

1. Start the AppSpace.
2. Run the `getcomponentdetail` command for a component of an application. For example:

```
BW_HOME\bin>bwadmin getcomponentdetail -d MyDomain -a MyAppSpace -n
MyAppNode MycomponentName tibco.bw.sample.MyApp.application 1.0
```

Admin UI

On the **Applications** page, the **Status** label displays the details of the component.

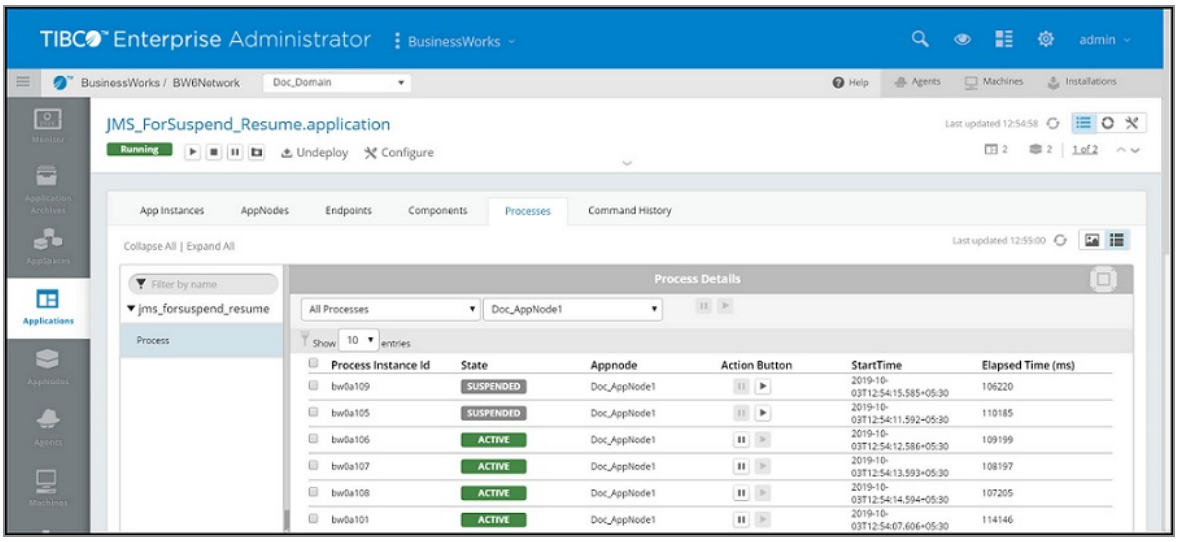
Suspending and Resuming Process Instances

Admin UI

Procedure

1. On the Application level 2 page, select **Processes** tab.

   Process Details view is opened.
2. To perform bulk operation on process instances, use filter based on process state (Active Processes, Suspended processes, All processes) as well as for AppNodes.



3. Use Select Columns filter to display additional columns such as Parent Process Name, Parent Process Instance Id, Main Process Name.

Backing Up and Restoring an Application

Backing up an application exports the current state of the specified application to a BWAdmin command file. The command file can be provided to BWAdmin to recreate the application state. Output can be compressed to a ZIP file with the `-zipped` option.

The default application profile is backed up with the application. Additional `*.substvar` files that contain application configurations can be created in the output folder.

A specific application version can be backed up by providing the `version` argument. If this argument is not specified, all application versions are backed up.

Procedure

1. To back up the current state of an application, perform the following actions:
    a. Open a terminal and navigate to *BW_HOME*`\bin`.
    b. Enter the `backup` command at the command line using `-s` option to identify the name of the destination file. Use the `-domain` and `appspace` options with the `application` argument. Provide the application version number to back up a specific version. By default, destination files are written to the current working directory.

This example backs the current state of an application to a command file named `app_backup.cmd`.

```
BW_HOME\bin>bwadmin backup -s app_backup.cmd -d Machine2Domain
-a AS1 application acme.acct.ap.application 1.0
```

2. To restore the application,
   a. Open a terminal and navigate to `BW_HOME\bin`.
   b. Enter the `bwadmin` command, providing the name of the backup command file. The following example recreates the state of the application *acme.acct.ap.application*. Because the application was deployed on backup, it is restored to the deployed state.

      ```
      BW_HOME\bin>bwadmin -f app_backup.cmd
      ```

      If you are restoring to a different location, you need to update the command file as follows:

      - The agent name points to `localhost` by default. You need to change this to the name of the machine that you are restoring to.
      - Update the domain home to point to the absolute path to the new location.
      - Update the path to the application archive (EAR) file to an absolute path.

   c. Use the `show applications` command with the `-domain` and `-appspace` options to verify the restore. If the BWAgent is not running, its status is listed as Unreachable.

Restoring the File System of an Archive

Restoring an archive restores the file system of the specified archive to the state of the datastore.

Before you begin

- The BWAgent must be running.

Note: Wildcards can be used to restore archives if archive names are not known.

Procedure

1. To restore the file system for an archive, open a terminal and navigate to `BW_HOME\bin`.
2. Enter the `restore` command, with the `-domain`, `-appspace`, and `appnode` options. Provide the `archive` argument specifying the name of the archive to restore or wildcards to restore all archives uploaded to domains in this agent.

   ```
   BW_HOME\bin>bwadmin restore -d Machine1Domain -a AS1 -n AN1 archive
   *.*
   ```

3. To verify the restore, check the file system. Open the *BW_HOME*`\domains` folder. Browse the folder and look for the archive in the *BW_HOME*`\domains\`*domain_name*`\archives` folder.

Restoring the File System of an Application

Restoring an application restores the file system of the specified application to the state of the datastore.

Before you begin

- The BWAgent must be running.

Note: Wildcards can be used to restore applications if application names are not known.

Procedure

1. To restore the file system of an application to the state of the datastore, open a terminal and navigate to *BW_HOME*`\bin`.
2. Enter the `restore` command at the command line, with the `-domain`, `-appspace`, and `-appnode` options. Provide the `application` argument with the name of the application or wildcards to restore all applications for this domain.

```
BW_HOME\bin>bwadmin restore -d Machine1Domain -a AS1 -n AN1
application *.*
```

3. To verify the restore, check the file system. Open the *BW_HOME*\domains folder. Look for the application in the *BW_HOME*\domains*domain_name*\appspaces*appspace_name*\apps folder.

Publishing APIs to TIBCO Cloud™ API Management

Follow these steps to publish application endpoints, or API endpoints, from the Admin UI to TIBCO Cloud API Management.

Before you begin

Complete the following tasks:

- Set up an API Management account.
- Register your application endpoint domain with API Management, and ensure that API Management can access it.
- Create the mashery.ini file:
    1. Open a text editor, and add the following properties with the correct values.

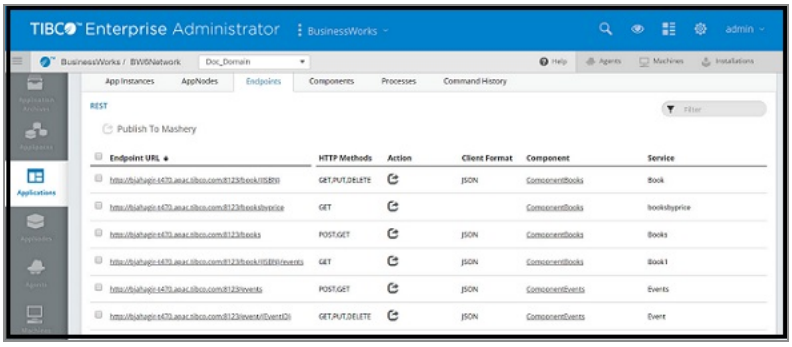
        ```
        bw.mashery.clientId=<clientId>
        bw.mashery.clientSecret=<clientSecret>
        bw.mashery.areaUuid=<areaUuid>
        bw.mashery.username=<username>
        bw.mashery.password=<password>
        bw.mashery.trafficManagerDomain=<trafficManagerDomain>
        bw.mashery.apiUrl=<https://api.HOST>
        ```

    2. Save the file as mashery.ini
    3. Add the mashery.ini file to *BW_HOME*/config.
- Edit the bwagent.ini file at *BW_HOME*/config/bwagent.ini, by adding the following line to the file:

    bw.mashery.config.file=../config/mashery.ini

Procedure

1. Start the application from the Admin UI.
2. Click the **Application** tab, and click the **Endpoints** tab.
3. From the **Endpoints** tab, select all, or individual, application endpoints.



> **Tip:** To select all application endpoints, select the **Select All** checkbox, located on the left of the Endpoint URL.

4. Click the Publish To Mashery icon, or the icon to publish the selected endpoints.

Result

Your API endpoints are published to TIBCO Cloud API Management.

For more information about managing your APIs from TIBCO Cloud API Management, see TIBCO Cloud™ API Management documentation.

Backing Up and Restoring from the Backup

The `backup` command backs up a specified runtime entity to a command file. You provide the command file as input to BWAdmin to recreate the environment.

The following runtime entities can be backed up and restored from a command file:

- Domain
- AppSpace
- AppNode
- Application

Runtime entities can be local or part of an agent network. Run this command frequently on all runtime entities, so that you always have a backup of your environment. Runtime entities created in local mode can only be restored when BWAdmin is in local mode. Runtime entities created in enterprise mode can only be restored when the BWAgent is running.

If you provide the same command file name on a subsequent backup, the existing command file is overwritten. Output can be compressed to a ZIP file with the `-zipped` option.

Note: The BWAdmin `backup` command and the BWAdmin `restore` command are not complimentary. The `restore` command requires a BWAgent and restores the file system to the state of the datastore. For information on the `restore` command, see Restoring the File System of Runtime Entities.

For more information about backing up and restoring from the backup, see:

- Backing Up and Restoring a Domain
- Backing Up and Restoring an Application
- Backing Up and Restoring an AppNode
- Backing Up and Restoring an AppSpace

Restoring the File System of Runtime Entities

The `restore` command restores the file system of a BWAgent or a runtime entity to the state of the datastore. If a machine goes offline and cannot be restarted, that environment can be recreated given the name of the BWAgent or the runtime entities.

The `restore` command can restore a BWAgent or a specified domain, AppSpace, AppNode, archive, or application. The command is only available when the BWAgent is running.

- When a BWAgent is restored, the file system for all runtime entities in the BWAgent datastore are restored.
- When a domain is restored, the file system for the specified domain and all contained runtime entities are restored.
- When an AppSpace is restored, the containing domain must exist. The file system for the specified AppSpace and all contained runtime entities are restored.

This pattern also applies to AppNodes, archives, and applications.

Note: The BWAdmin `backup` command and the BWAdmin `restore` command are not complimentary. The `backup` command backs up a runtime entity to a command file. For information on the `backup` command, see Backing Up and Restoring from the Backup.

For more information about restoring the file system, see the following topics:

- Restoring the File System of a BWAgent
- Restoring the File System of a Domain
- Restoring the File System of an AppSpace
- Restoring the File System of an AppNode
- Restoring the File System of an Archive
- Restoring the File System of an Application

Smart Engine

ActiveMatrix BusinessWorks collects engine data on an AppNode. Based on the engine data collected, it generates HTML reports and provides analysis and recommendations for improving your application's performance.

Generating Reports for Engine Data

You can generate various reports under some conditions such as increased memory usage, high CPU usage, and more live threads for certain time.

By default, ActiveMatrix BusinessWorks generates reports at `{BW_HOME}\<product_version>\Reports`. You can change the report location by setting the `bw.smartengine.report.path={path_to_report_folder}` property in the AppNode's `config.ini` file. You can also maintain a history or reports.

Before you begin

- Set the following property in the AppNode's `config.ini` file.

  ```
  bw.smartengine.enabled=true
  ```

  You can also enable the smart engine feature dynamically by using the following REST API:

  ```
  http://<host>:<appnode_
  port>/monitor/systemproperties/enableSmartEngine
  ```

  Note: The default endpoint for the Smart Engine is `http://localhost:8090/monitor` where, `8090` is the AppNode port.

Procedure

1. To get the application statistics in reports, set the `bw.smartengine.appStatistics.enabled` property to `true` in the AppNode's `config.ini` file. You can also enable the application statistics dynamically by using the following REST API:

   ```
   http://<host>:<appnode_
   port>/monitor/systemproperties/enableSmartEngine?bw.smartengine.app
   Statistics.enabled=true
   ```

   Warning: You may observe performance degradation after setting the property.

2. Based on your requirements to get data in the report, several triggers are available. For more information, see the list of available Triggers.

3. To keep a specific number of reports for each type of performance use case at `{BW_HOME}\<product_version>\Reports` location, set the following properties in an AppNode's `config.ini` file:

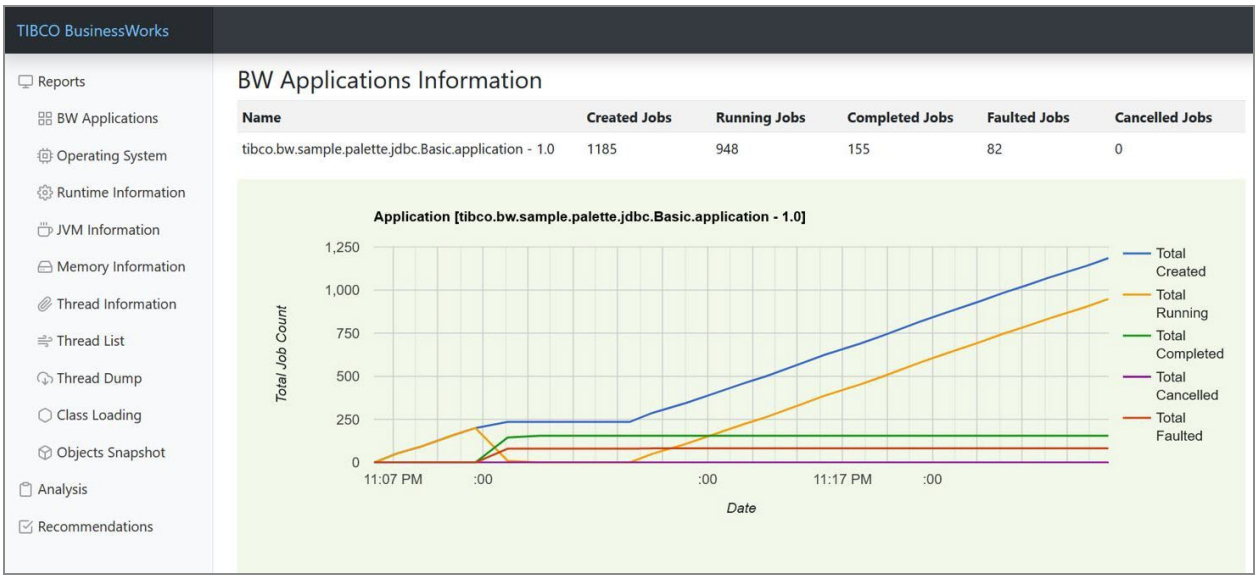
   `bw.smartengine.keepRecentReports.enabled=true`

   `bw.smartengine.keepRecentReports.size=5`

   By default, the smart engine stores the previous five reports for each performance use case.

Result

The report is stored at your specified location in the `.zip` format. The `.zip` file contains a report in an HTML format. The report has the following layout:



The HTML report has the following sections:

Section	Description
BW Applications	This section shows the ActiveMatrix BusinessWorks applications in a table. Each row shows the number of jobs for the ActiveMatrix BusinessWorks processes in an application, such as Created Jobs, Running Jobs, Completed Jobs, Faulted Jobs, and Canceled Jobs. After the ActiveMatrix BusinessWorks applications table, line charts are shown for each application, such as Total Job Count chart and New Job Count chart. When an application has incoming HTTP requests, Total HTTP Connector Calls chart, and New HTTP Connector Calls chart are shown. For each ActiveMatrix BusinessWorks application, the ActiveMatrix BusinessWorks processes in the ActiveMatrix BusinessWorks application are shown in a table. Each row shows the number of jobs for a ActiveMatrix BusinessWorks process, such as Created, Completed, Faulted, and Suspended. After the ActiveMatrix BusinessWorks processes table, line charts are shown for each ActiveMatrix BusinessWorks process, such as Total Job Count chart and New Job Count chart. For each ActiveMatrix BusinessWorks process, the ActiveMatrix BusinessWorks activities in the ActiveMatrix BusinessWorks process are shown in a table.

Section	Description
	Each row shows the runtime information of an activity such as: • Recent Status • Executed • Faulted • Recent Elapsed Time (ms) • Min Elapsed Time (ms) • Max Elapsed Time (ms) • Total Elapsed Time • Recent Activity Output Memory (bytes) • Min Activity Output Memory (bytes) • Max Activity Output Memory (bytes) The processes and activities statistics data is available when the application statistics feature is enabled (for example, `bw.smartengine.appStatistics.enabled=true`).
Operating System	This section shows the operating system information in a table, such as: • OS Name • OS Version • OS Architecture • Available Processors • Committed Virtual Memory • Free Physical Memory • Total Physical Memory • Free Swap Space • Total Swap Space

Section	Description
	• JVM Process CPU Time • JVM CPU Load • System CPU Load • System Load Average After this table, line charts are shown, such as: • Free Physical Memory and Free Swap Space chart • JVM CPU Load and System CPU Load chart • System Load Average chart
Runtime Information	This section shows the runtime JVM information in a table, such as: • Process Name • Spec Name • Spec Vendor • Spec Version • VM Name • VM Version • VM Vendor • Management Spec Version • Start Time • Up Time • Class Path • Library Path • Input Arguments • System Properties

Section	Description
JVM Information	This section shows the overall JVM information in a table such as: • PID • Java Vendor • Java Name • Java Version • OS User • CPU Load • Up Time • GC Time • GC Count • GC Load • Max Heap • Used Heap • Used Non-Heap • Total Loaded Class Count • Thread Count • Peak Thread Count • Total Started Thread Count After this table, Top Threads information is shown in a table. Each row shows the data of a thread, such as TID, Name, State, Thread CPU Usage(%), Thread Total CPU Usage(%) and Blocked Thread. After that, the Top Methods information is shown in a table. Each row shows the data of a method, such as Class Name, Method Name, and Total CPU Time(ms).
Memory Information	This section shows the JVM memory information in a table, such as: • Max Heap Size

Section	Description
	• Committed Heap Size • Init Heap Size • Used Heap Size • Max Non-Heap Size • Committed Non-Heap Size • Init Non-Heap Size • Used Non-Heap Size After the table, line charts are shown, such as Heap Memory Usage chart and Non-Heap Memory Usage chart.
Thread Information	This section shows the overall JVM threads information in a table, such as: • Thread Count • Daemon Thread Count • Peak Thread Count • Total Started Thread Count • Current Thread CPU Time • Current Thread User Time After this table, a Thread State Count table is shown. Each row shows the number of threads in a Thread state, such New, Runnable, Blocked, Waiting, and Timed Waiting. After that, line charts are shown, such as JVM Thread Count chart and JVM Thread State Count chart.
Thread List	This section shows the JVM threads in a table. Each row shows the data of a thread, such as TID, Name, State, CPU Time(ms), and Allocated Heap Size.
Thread Dump	This section shows the JVM threads dump in a table. Each row shows

Section	Description
	the thread dump of a thread, such as TID, Thread Name, Thread State, Thread Allocated Heap, and Stack Trace.
Class Loading	This section shows the JVM class loading information in a table, such as Loaded Class Count, Total Loaded Class Count, and Unloaded Class Count. After the table, a line chart of Classes Count is shown.
Objects Snapshot	This section shows the JVM objects in a table. Each row shows the data of an object, such as number of Instances, Allocated Heap Size, and Class name.
Analysis	This section shows the analysis of various performance use cases. When the triggers are evaluated and a trigger condition is met for a performance use case, a corresponding analysis is provided and shown in the report.
Recommendations	This section shows the recommendations for various performance use cases. When the triggers are evaluated and a trigger condition is met for a performance use case, related recommendations are provided by the corresponding recommendation providers and shown in the report.

What to do next

Based on the recommendations, modify the properties and redeploy the application for better performance.

Triggers

You can populate the data in a report based on certain conditions. When those conditions are met, the trigger is executed. Based on your requirements, you can modify threshold values by using REST APIs.

Note: Use `http://<host>:<appnode_port>/monitor` as a base URL for all the REST APIs provided where, `http://localhost:8090/monitor` is the default endpoint. Here, `8090` is the AppNode port.

The following triggers are available:

High CPU Trigger

ID	bw.montr.trigger.HighCPUTrigger
Threshold	highCpuThresholdPercent: 80 highCpuDurationMins: 5
Description	The trigger measures the high CPU usage situation. The trigger conditions are met when CPU usage is equal to or greater than 80% and the situation has lasted for more than (including) 5 minutes.

High Memory Trigger

ID	bw.montr.trigger.HighMemoryTrigger
Threshold	highMemoryThresholdPercent: 80 highMemoryDurationMins: 5
Description	The trigger measures the high memory usage situation. The trigger conditions are met when memory usage is equal to or greater than 80% and the situation has lasted for more than (including) 5 minutes.

Out of Memory Trigger

ID	bw.montr.trigger.OutOfMemoryTrigger
Threshold	outOfMemoryThresholdPercent: 95

Description	The trigger measures a very high memory usage situation (very close to out of memory). The trigger condition is met when memory usage is equal to or greater than 95%.

High Live Threads Trigger

ID	bw.montr.trigger.HighLiveThreadsTrigger
Threshold	highLiveThreadsThreshold: 500 highLiveThreadsDurationMins: 5
Description	The trigger measures the high number of live threads situation. The trigger conditions are met when the number of live threads (including both daemon and non-daemon threads) is equal to or greater than 500 and the situation has lasted for more than (including) 5 minutes.

High JMS Queue Pending Messages Trending Trigger

ID	bw.sharedresource.trigger.HighQueuePendingMessagesTrendingTriggerAction
Threshold	queuePendingMessagesCountMinValueThreshold: 1000 queuePendingMessagesTrendingPercentThreshold: 300 queuePendingMessagesDurationMinutesThreshold: 5
Description	The trigger measures the delay of processing JMS messages situation by checking the trending of pending messages in JMS queues that are accessed by activities in each ActiveMatrix BusinessWorks application. The trigger conditions are met when the pending messages in a JMS queue have increased by more than (including) 300 percent in the recent 5 minutes with a minimum pending messages of 1000.

High JMS Queue Pending Messages Count Trigger

ID	bw.sharedresource.trigger.HighQueuePendingMessagesCountTriggerAction
Threshold	queuePendingMessagesCountThreshold: 10000 queuePendingMessagesDurationMinutesThreshold: 5
Description	The trigger measures the delay of processing JMS messages situation by checking the number of pending messages in JMS queues that are accessed by activities in each ActiveMatrix BusinessWorks application. The trigger conditions are met when the number of pending messages in a JMS queue is equal to or greater than 10000 and the situation has lasted for more than (including) 5 minutes.

HTTP Connector Acceptor Thread Count Threshold Trigger

ID	bw.sharedresource.trigger.HttpConnectorAcceptorThreadCountThresholdTriggerAction
Threshold	-
Description	The default value of the HTTP Acceptor Thread Count Configuration on the HTTP Connector Shared Resource is 1. Jetty provides a formula for the maximum number of acceptor threads that can be allocated based on the available machine processors. The trigger checks the under-utilized acceptor threads that is the trigger condition is met when the configured value is less than the maximum allowed acceptor thread count value.

HTTP Connector Executor Threadpool Utilization Threshold Trigger

ID	bw.sharedresource.trigger.HttpConnectorExecutorThreadpoolUtilizationThresholdTriggerAction
Threshold	executorThreadpoolUtilizationThreshold: 85.0

	highThreadpoolUtilizationDurationMinutesThreshold: 5.0
Description	The trigger measures the thread pool utilization while processing the incoming HTTP requests. The trigger conditions are met when the thread pool utilization is more than (including) 85% over the span of 5 minutes by default.

HTTP Connector Queue Utilization Threshold Trigger

ID	bw.sharedresource.trigger.HttpConnectorQueueUtilizationThresholdTriggerAction
Threshold	connectorThreadpoolQueueUtilizationThreshold: 85.0
Description	The Blocking Queue size for the Jetty server in ActiveMatrix BusinessWorks 6.x can be set using the System property `bw.engine.http.jetty.blockingQueueSize=<Integer Value>`. The trigger measures this Jetty Blocking queue utilization percentage. The trigger conditions are met when the blocking queue size is more than (including) 85% full by default.

Triggers REST API

Considering `8090` as the AppNode port, this section describes the following Triggers REST APIs:

- http://localhost:8090/monitor/triggers
- http://localhost:8090/monitor/triggers/{triggerId}/properties

http://localhost:8090/monitor/triggers

Method	GET
Description	Get a list of triggers of the smart engine.

(Continued)

Path Parameters	None
Query Parameters	None
Header Parameters	None
Output	• Code = 200 Message = "Returns a list of triggers." • Code = 503 Message = "Internal Server Error".

http://localhost:8090/monitor/triggers/{triggerId}/properties

Method	PUT
Description	Update the properties of a trigger.
Path Parameters	• Parameter: triggerId • Type: String(required) • Description: The id of a trigger
Query Parameters	None
Header Parameters	None
Body Parameters	{ "{propertyName1}": {propertyValue1}, "{propertyName2}": {propertyValue2}, ... "{propertyNameN}": {propertyValueN} }
Output	• Code = 200

(Continued)

	Message = "Trigger's properties are updated." • Code = 503 Message = "Internal Server Error".
Sample Output	{ "highIdleTimeoutPerMinuteThreshold": 60, "highIdleTimeoutDurationMinutesThreshold": 5 } { "code": "200", "message": "Trigger's properties are updated.", "status": "success" }

Reports REST API

Considering `8090` as the AppNode port, this section describes the following Reports REST APIs:

- http://localhost:8090/monitor/reports
- http://localhost:8090/monitor/reports/generate
- http://localhost:8090/monitor/reports/{reportId}/download
- http://localhost:8090/monitor/reports/{reportId}/delete
- http://localhost:8090/monitor/reports/deleteall

http://localhost:8090/monitor/reports

Method	GET
Description	Get a list of reports
Path Parameters	None
Query Parameters	None
Header	None

(Continued)

Parameters	
Output	• Code = 200 Message = "Returns a list of reports." • Code = 503 Message = "Internal Server Error".
Sample Output	[{ "id": "Report-2021-08-26T12-56-50-0700", "date": "2021-08-26 12:56:50" }, { "id": "Report-2021-08-26T16-42-14-0700", "date": "2021-08-26 16:42:14" }]

http://localhost:8090/monitor/reports/generate

Method	GET
Description	Generate a report manually
Path Parameters	None
Query Parameters	• Parameter: engineData • Type: Boolean(Optional) • Description: Whether to generate a APPNODE_DATA file in the report zip file. By default, the value is false.
Header Parameters	None

(Continued)

Output	• Code = 200 Message = "Reports are generated." • Code = 503 Message = "Internal Server Error".

Sample Output	{
	"code": "200",
	"message": "Reports are generated.",
	"status": "success"
	}

http://localhost:8090/monitor/reports/{reportId}/download

Method	GET
Description	Download a report `.zip` file.
Path Parameters	• Parameter: reportId • Type: String (required) • Description: The id of a report
Query Parameters	None
Header Parameters	None
Output	• Code = 200 Message = "Download a report zip file." • Code = 503 Message = "Internal Server Error".

(Continued)

Sample Output	{
	"code": "200",
	"message": "Reports are generated.",
	"status": "success"
	}

http://localhost:8090/monitor/reports/{reportId}/delete

Method	GET
Description	Delete a report.
Path Parameters	• Parameter: reportId • Type: String (required) • Description: The id of a report
Query Parameters	None
Header Parameters	None
Output	• Code = 200 Message = "Delete a report" • Code = 503 Message = "Internal Server Error"

Sample Output	{
	"code": "200",
	"message": "Report is deleted",
	"status": "success"
	}

http://localhost:8090/monitor/reports/deleteall

Method	GET
Description	Delete all reports.
Path Parameters	None
Query Parameters	None
Header Parameters	None
Output	• Code = 200 Message = "Reports are deleted" • Code = 503 Message = "Internal Server Error"

Sample Output	{
	"code": "200",
	"message": "Reports are deleted",
	"status": "success"
	}

Properties REST API

Considering `8090` as the AppNode port, this section describes the following Properties REST APIs:

- http://localhost:8090/monitor/systemproperties/enableSmartEngine?bw.smartengine.appStatistics.enabled=true
- http://localhost:8090/monitor/systemproperties/disableSmartEngine

http://localhost:8090/monitor/systemproperties/enableSmartEngine?bw.smartengine.appStatistics.enabled=true

Method	GET
Description	Enable smart engine
Path Parameters	None
Query Parameters	• Parameter: bw.smartengine.appStatistics.enabled • Type: Boolean (Optional) • Description: Whether to enable application statistics. Default value is false.
Header Parameters	None
Output	• Code = 200 Message = "System property is set with old value and new value." • Code = 503 Message = "Internal Server Error".

Sample Output	{
	"code": "200",
	"message": "Smart engine is enabled.",
	"status": "success"
	}
	{
	"code": "200",
	"message": "Smart engine (with application statistics) is enabled.",
	"status": "success"
	}

http://localhost:8090/monitor/systemproperties/disableSmartEngine

Method	GET
Description	Disable smart engine
Path Parameters	None
Query Parameters	None
Header Parameters	None
Output	• Code = 200 Message = "System property is set with old value and new value." • Code = 503 Message = "Internal Server Error".

Sample Output	{
	"code": "200",
	"message": "Smart engine is disabled.",
	"status": "success"
	}

Debugging

In the event of an error, the software provides detailed messages that can help you trace through to the cause of an issue. You can adjust logging levels as needed to capture different granularity of messages for different loggers.

Messages returned by ActiveMatrix BusinessWorks are categorized by component and by error code within component. The following table shows the components of the software that return messages:

Component ID	Description
BX	ActiveMatrix BusinessWorks Engine Layer
PVM	ActiveMatrix BusinessWorks Engine Layer
TIBCO-BW-ADMIN	ActiveMatrix BusinessWorks Administrator
TIBCO-BW-ADMIN-CLI	ActiveMatrix BusinessWorks Administrator Command Line Interface
TIBCO-BW-BINDING-REST	ActiveMatrix BusinessWorks REST Binding
TIBCO-BW-BINDING-SOAP	ActiveMatrix BusinessWorks SOAP Binding
TIBCO-BW-CORE	ActiveMatrix BusinessWorks Engine Layer
TIBCO-BW-FRWK	ActiveMatrix BusinessWorks Framework
TIBCO-BW-PALETTE	ActiveMatrix BusinessWorks Palette Layer
TIBCO-BW-PALETTE-<*PaletteName*>	ActiveMatrix BusinessWorks Palette specific activity implementation
TIBCO-BW-SR	ActiveMatrix BusinessWorks Shared Resource API Layer
TIBCO-BW-SR-<*UniqueName*>	ActiveMatrix BusinessWorks specific Shared Resource

Component ID	Description
	implementation
TIBCO-BW-STATS	ActiveMatrix BusinessWorks Stats Collector
TIBCO-THOR-FRWK	Thor Framework

Note: The Engine layers with component IDs BX and PVM do not follow this convention.

The following tables identifies the log levels used by the software and the corresponding error code range:

Log Level	Error Code
TRACE	100001 - 109999
DEBUG	200001 - 209999
INFO	300001 - 309999
WARN	400001 - 409999
ERROR	Errors can be indicated by one of the following error code ranges: • 500001 - 509999 • 600001 and higher • 0xxxxx **Note:** • Error codes 600001 and higher indicate exceptions in the execution and are always associated with a 5xxxxx error code, which can be traced in the log file. • Error codes starting with 0xxxxx indicate internal errors. Contact TIBCO Support for possible resolution or a workaround.

If you encounter an error that does not start with a 0, change the logging level of the AppNode and BWAgent loggers to DEBUG. Try to recreate the scenario and examine the log file to try and trace the issue. The error messages are detailed and can help you understand the chain of events that led up to the issue. For more information on log files and log file configuration, see Logging in the *TIBCO ActiveMatrix BusinessWorks™ Administration* guide.

The following sections in the *TIBCO ActiveMatrix BusinessWorks™ Concepts* guide list issues you might encounter, along with possible resolutions:

- Troubleshooting BWAgent Issues
- Troubleshooting Runtime Entities Issues
- Troubleshooting Archive Issues
- Troubleshooting Application Issues
- Troubleshooting TIBCO Enterprise Administrator Integration Issues

Troubleshooting BWAgent Issues

Some BWAgent issues and possible resolutions are listed below. This list is not complete but provides examples of messages that might be returned.

For a complete list of error codes, see the *TIBCO ActiveMatrix BusinessWorks™ Error Codes* guide.

Issue	Message	Resolution
When registering a BWAgent to a domain or AppSpace, or unregistering the AppSpace the BWAgent cannot be registered.	TIBCO-BW-ADMIN-CLI-500132: Failed to unregister BWAgent [*bwagent* from the AppSpace [*AppSpace*]. <CausedBy> TIBCO-BW-ADMIN-500004: Error	Verify the BWAgent name. Check that the BWAgent on the remote machine is running.

Issue	Message	Resolution
	invoking the [*methodName*] method on the agent [*Agent*], The BWAgent [*Agent*] on the remote machine is not running.	
If you are unable to enable or disable the autoregistration feature from the command line after running the `enableautoregistration` utility or the `disableautoregistration` utility.	TIBCO-BW-AGENT-500004: Error invoking [method] method on the agent [BWAgent]. The BWAgent [`bwagent`] on the remote machine is not running.	Check if BWAgent is running, and if it is, verify its name.
The BWAgent could not start because the mode is not set to enterprise.	TIBCO-BW-AGENT-500006: Cannot start the agent. The admin mode in `bwagent.ini` is not configured for enterprise mode. Check your configuration and restart.	Open a terminal and type the following command at the command line: `bwadmin mode enterprise` Restart the BWAgent.
A BWAdmin command could not be completed.	TIBCO-BW-ADMIN-500008: Error in initializing data manager, TIBCO-BW-ADMIN-PRSTNC-500001: Connection to	The first message indicates that the BWAgent is not running. Start the BWAgent. The second message is displayed when BWAdmin is configured for enterprise mode and the command could not be completed

Issue	Message	Resolution
	BWAgent failed. Or TIBCO-BW-ADMIN-CLI-500006: Failed to initialize transport, TIBCO-BW-ADMIN-PRSTNC-500001: Connection to BWAgent failed.	due to a failed BWAgent connection.
The BWAgent could not start due to an error with datastore initialization.	TIBCO-BW-AGENT-500009: Failed to start agent due to an error in initializing data store, reason: *Reason*	The datastore is written to the *BW_HOME*\domains\.datastore folder. Verify the following: * The specified folder exists. * The bw.agent.technology.as.dataStoreLocation property in the *BW_HOME*\config\bwagent.ini file points to the datastore folder name. * The bw.agent.technology.as.role property is set to server. You might also see this message if there is an issue with the bw.agent.technology.as.dataStoreLocation property setting in the *BW_HOME*\config\bwagent.ini file. Make sure that the TCP protocol is specified only for the first URL in the string; not for subsequent URLs.
When starting a BWAgent that is configured as part	Unable to resolve network	Check the setting of the bw.agent.technology.as.dataStoreLocation property in the *BW_*

Issue	Message	Resolution
of an agent network, the URL of another agent in the network could not be found.	specification ('*bwagent*')	*HOME*\config\bwagent.ini file. Verify that: • The URL is specified as either IP address and port or host name and port, in the format: *IP_address/hostname*:*port* • There are no typos in the URL or port number. • A semicolon separator is used between URLs.
When starting a BWAgent that is configured as part of an agent network, the BWAgent starts but does not indicate that the agent has joined the network.	N/A	Check that the setting of the bw.agent.network.name property in the *BW_HOME*\config\bwagent.ini file is the same as the setting in other the configuration file for other BWAgents in the network.
When starting a BWAgent that is configured as part of a BWAgent network, and the BWAgent is not configured for the network. Warnings are displayed.	There are [2] agents in the BWAgent group that store data. However, the property "minSeederCount" in the bwagent.ini file is set to [1], see the bwagent.ini file or the documentation and choose an appropriate value. There are [2]	For more information, see comments in the *BW_HOME*\config\bwagent.ini file.

Issue	Message	Resolution
	agents in the BWAgent group that store data. However, the property "quorumSize" in the `bwagent.ini` file is set to [1], see the `bwagent.ini` file or the documentation and choose an appropriate value.	
The TEA agent could not be registered.	TIBCO-BW-ADMIN-500504: Failed to register TEA Agent [*teaagent*] with TEA server [`http://host:` *port*], TEA Agent registration failed, TIBCO-BW-TEAAGENT-500300: Failed to register BW TEA agent [*teaagent*] with TEA server, <CausedBy> Unable to register agent with name [*teaagent*]'	Verify the URL to the TEA agent. Check that the TEA agent is running.

Whenever you initiate any operation such as create domain, create AppNode, the BWAgent generates a unique ID for that operation and that ID persists till that operation is complete.

For example, when you create a domain, you get the following entries in the `bwagent.log` file:

```
INFO  [<thread_number>] <fb587904-ab0c-4403-b682-da809ac57b96>
c.t.b.t.m.d.u.DomainLifecycleCommand - Creating domain [<domain_name>]
at default location
 INFO  [<thread_number>] <fb587904-ab0c-4403-b682-da809ac57b96> bw.audit
- create -agent <agent_name> -domainHome <domain_home> domain <domain_
name>
INFO  [<thread_number>] <fb587904-ab0c-4403-b682-da809ac57b96>
c.t.b.t.m.d.u.DomainLifecycleCommand - TIBCO-BW-ADMIN-300100: Created
the domain [<domain_name>]
```

To retrieve the unique id of the operation run, follow the steps:

BWAdmin Command Line

1. Open *BW_HOME*`/bin/bwadmin-logback.xml` in a text editor. Change the ROOT level setting at the end of the file as needed.

   ```
   <root level="DEBUG">
       <appender-ref ref="STDOUT" />
       <appender-ref ref="FILE" />
   ```

2. Open the *BW_HOME*`/bin/bwagent-logback.xml` file in a text editor. Change the ROOT level setting at the end of the file as needed.

   ```
   <root level="DEBUG">
       <appender-ref ref="STDOUT" />
       <appender-ref ref="FILE" />
   ```

3. Perform the operation such as Create Domain.
4. Open the *BW_HOME*`/logs/bwadmin.log` file and retrieve the unique ID. The unique id is present at the beginning of an operation.

   ```
    2020-12-08 15:32:10.412 DEBUG [main]
   <5a83a476-67f1-4bc7-ba49-70451c1bf320> com.tibco.thor.frwk -
   Starting Execution for command
   [com.tibco.bw.thor.admin.cli.commands.CreateEntityCommand]
   at Time [timestamp] with
   ID[5a83a476-67f1-4bc7-ba49-70451c1bf320]
   ```

5. You can verify the logs associated with the operation in the `bwagent.log` file using the same unique id of the operation as per the step 4.

BWAgent REST API

1. Open the *BW_HOME*`/bin/bwagent-logback.xml` file in a text editor. Change the ROOT level setting at the end of the file as needed.

```
<root level="DEBUG">
    <appender-ref ref="STDOUT" />
    <appender-ref ref="FILE" />
```

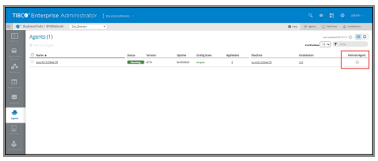
2. Perform the operation such as Create Domain.
3. On receiving the response, you get the operation ID in the response headers.
4. You can verify the logs associated with the operation in the `bwagent.log` file using the operation id obtained in the step 3.

Troubleshooting Runtime Entity Issues

Some runtime entity issues and possible resolutions are listed below. This list is not complete but provides examples of messages that might be returned.

For a complete list of error codes, see the *TIBCO ActiveMatrix BusinessWorks™ Error Codes* guide.

Issue	Message	Resolution
The status of AppNode or AppInstance is not shown correctly.	N/A	Use the **Refresh Agent** button. • On Agent's page on Admin UI. In case of domain-specific status update, the status of all AppNodes and AppInstances within that domain is updated.

Issue	Message	Resolution
		 In case of a non-domain agent page, the status of all AppNodes and AppInstances running on that agent is updated. • Form Admin CLI with the following command: `BW_HOME\bin>bwadmin refresh`
The specified runtime entity could not be created as it contains invalid characters or contains over 100 characters.	TIBCO-BW-ADMIN-CLI-500501: Name contains invalid characters and does not comply with naming conventions. Valid characters are upper and lower case characters of the alphabet as well as digits, '.' and '-'. Or TIBCO-BW-ADMIN-CLI-500502: Name length exceeds 100 characters and it can be truncated to satisfy the length limit.	Create the runtime entity with valid characters: • `A-Z` • `a-z` • `0-9` • - (hyphen) • _ (underscore) Illegal characters are stripped from the name. The maximum length of a runtime entity name is 100 characters. If the maximum length is exceeded, the entity name is shortened to 100 characters.

Issue	Message	Resolution
The specified domain could not be created. It exists.	TIBCO-BW-ADMIN-CLI-500102: Failed to create Domain [*Domain*], TIBCO-BW-ADMIN-500101: Domain [*Domain*] exists, check this and retry.	The specified domain exists. Domain names must be unique; enter a different name. To view existing domains, use the `show domains` command.
The specified command could not be completed. The domain home folder specified could not be found.	TIBCO-BW-ADMIN-CLI-500103: Domain home folder [*Path*] does not exist. Verify if the folder is present and use a forward slash in the folder path.	Check that the *`BW_HOME`*`\domains` folder exists. If it does exist, make sure that forward slashes are specified (both Windows and Unix).
The specified domain could not be deleted.	TIBCO-BW-ADMIN-CLI-500104: Failed to delete Domain [*Domain*], *Reason*	The specified domain may have AppSpaces associated with it. If this is the case, the following message is displayed: TIBCO-BW-ADMIN-500109: The Domain [*Domain*] has AppSpaces associated with it. Use the `-force` option to override. Use the `-force` option with the `delete` command to delete the domain and all contained runtime entities.
The minimum number of AppNodes for an AppSpace has to be at least 1.	TIBCO-BW-ADMIN-CLI-500218: The minNodes configuration value for an AppSpace has to be at least 1.	The `minNodes` value for an AppSpace has to be set to an integer value greater than 0. To create an AppSpace with 1 AppNode, the `minNodes` option is not required.
The minimum	TIBCO-BW-ADMIN-CLI-500219: The	The `minNodes` value for an

Issue	Message	Resolution
number of AppNodes is invalid. It has to be an integer value greater than 0.	minNodes configuration argument is invalid. Valid arguments are Integer values greater than 0.	AppSpace has to be set to an integer value greater than 0.
The AppSpace could not be found in the specified domain.	TIBCO-BW-ADMIN-CLI-500201: AppSpace [*AppSpace*] not found in Domain [*Domain*]	The specified AppSpace does not exist in the specified domain. Check the value for typos. Use the `show appspaces` command with the `-domain` option to view AppSpaces in the domain.
The specified AppSpace in the specified domain could not be created. It exists.	TIBCO-BW-ADMIN-CLI-500203: Failed to create AppSpace [*AppSpace*] in Domain [*Domain*]. TIBCO-BW-ADMIN-500202: The AppSpace [*AppSpace*] is already present in the Domain [*Domain*].	The specified AppSpace has already been created. Use the `show appspaces` command with the `-domain` option to view AppSpaces in the domain.
The specified AppSpace exists for the specified BWAgent.	TIBCO-BW-ADMIN-CLI-500204: AppSpace [*AppSpace*] exists with BWAgent [*bwagent*]	The specified AppSpace has already been expanded to the specified BWAgent.
The specified AppSpace in the specified domain could not be started.	TIBCO-BW-ADMIN-CLI-500210: AppSpace [*AppSpace*] in Domain [*Domain*] could not be started, *Reason*	The specified AppSpace may not have associated AppNodes or it might be on a BWAgent that is not reachable. The following message is displayed if there are no contained AppNodes: TIBCO-BW-ADMIN-500210: AppSpace [*AppSpace*] in Domain [*Domain*] did not start completely,

Issue	Message	Resolution
		status is Degraded. An AppSpace can only be started if it contains at least one AppNode. Use the `show appnodes` command with the `-appspace` and `-domain` options to check for AppNodes. If no AppNodes exist, create at least one and try to start the AppSpace again. If the minimum number of AppNodes was specified when the AppSpace was created, that minimum number of AppNodes must exist. A message similar to the following is displayed if the AppSpace is on an unreachable BWAgent: TIBCO-BW-ADMIN-500210: AppSpace [*AppSpace*] in Domain [*Domain*] did not start completely, status is Stopped. The machine might be down or the BWAgent might not be running.
The specified AppSpace in the specified domain could not be deleted.	TIBCO-BW-ADMIN-CLI-500205: Failed to delete AppSpace [*AppSpace*] from Domain [*Domain*], *Reason*	The specified AppSpace may have AppNodes associated with it or it may be scaled across BWAgents. If it has associated AppNodes, the following message is displayed: TIBCO-BW-ADMIN-500216: AppSpace [*AppSpace*] has AppNodes associated with it. Delete the AppNodes first and

Issue	Message	Resolution
		retry or use the `-force` option to override. If the AppSpace is scaled across machines, the following message is displayed: TIBCO-BW-ADMIN-500220: AppSpace [*AppSpace*] is scaled across multiple BW Agents. Cannot be deleted. Use the `-force` option to override. In both cases, either delete the contained AppNodes or use the `-force` option with the `delete` command to delete the AppSpace and all contained runtime entities.
When creating an AppNode, the HTTP Port value is required.	httpPort is a mandatory argument for creating an AppNode.	Create the AppNode again with the `-httpPort` option. The port must be unique for each AppNode on the machine.
The specified AppNode in the specified AppSpace and domain could not be created. It exists.	TIBCO-BW-ADMIN-CLI-500302: Failed to create AppNode [*AppNode*] in AppSpace [*AppSpace*] in Domain [*Domain*], TIBCO-BW-ADMIN-500301: The AppNode [*AppNode*] exists in the AppSpace [*AppSpace*] Domain [*Domain*].	The specified AppNode has already been created. Use the `show appnodes` command with the `-appspace` and `-domain` options to view AppNodes.
The specified AppNode in the specified AppSpace and domain could not be started.	TIBCO-BW-ADMIN-CLI-500304: AppNode [*AppNode*] in Domain [*Domain*] did not start, *Reason*	The specified AppNode may not exist. In this case, the following error is displayed: TIBCO-BW-ADMIN-500300: The AppNode [*AppNode*] does not exist in AppSpace [*AppSpace*] and

Issue	Message	Resolution
		Domain [*Domain*]. If the BWAgent is in a network, the AppSpace might be on a BWAgent that is not reachable. The machine might be down or the BWAgent might not be running. In this case, the following error is displayed: TIBCO-BW-ADMIN-500004: Error invoking [startappnode] method on the agent [*agent*], The BWAgent [*agent*] on the remote machine is not running. Start the BWAgent on the remote machine and start the AppNode again.
The specified AppNode in the specified AppSpace and domain could not be deleted.	TIBCO-BW-ADMIN-CLI-500306: Failed to delete AppNode [*AppNode*] in AppSpace [*AppSpace*] in Domain [*Domain*]	The specified AppNode may be running. If this is the case, the following message is displayed: TIBCO-BW-ADMIN-500314: The AppNode [*AppNode*] is still in [Running] state. Please stop the AppNode first or use the -force option. Stop the AppNode and delete it or use the `-force` option with the `delete` command. If the BWAgent is in a network, the AppNode might be on a BWAgent that is not reachable. The machine might be down or the BWAgent might not be running. In this case, the following error is displayed:

Issue	Message	Resolution
		TIBCO-BW-ADMIN-500306: Failed to delete AppNode [*AppNode*] in AppSpace [*AppSpace*] in Domain [*Domain*], The BWAgent [*agent*] on the remote machine is not running.
When the AppNode is started, the specified HTTP port cannot be allocated.	TIBCO-THOR-FRWK-500300: Eclipse Jetty server bundle has reported an error that it cannot allocate the HTTP management port [*port*]. Shutting down the AppNode.	An AppNode (or a JVM) is already running with the specified port. Stop that AppNode or process and restart the AppNode.
The OSGi console could not be enabled on the specified AppNode.	TIBCO-BW-ADMIN-CLI-500314: Failed to enable console on AppNode [*AppNode*] in AppSpace [*AppSpace*] in Domain [*Domain*], *Reason*	The OSGi console could not be enabled for the specified AppNode. The AppNode is not running or is not reachable. In this case, the following additional message is displayed: TIBCO-BW-ADMIN-500309: Failed to enable console on AppNode [*AppNode*] in AppSpace [*AppSpace*] in Domain [*Domain*] Make sure the AppNode is running and try again.
The OSGi console could not be disabled on the specified AppNode.	TIBCO-BW-ADMIN-CLI-500315: Failed to disable console on AppNode [*AppNode*] in AppSpace [*AppSpace*] in Domain [*Domain*], *Reason*	The OSGi console could not be disabled for the specified AppNode. The AppNode is not running or is not reachable. In this case, the following additional message is displayed: TIBCO-BW-ADMIN-500310: Failed to disable console on AppNode

Issue	Message	Resolution
		[*AppNode*] in AppSpace [*AppSpace*] in Domain [*Domain*] Make sure the AppNode is running and try again.
The debug port on the AppNode could not be enabled.	TIBCO-BW-ADMIN-CLI-500440: Failed to enable debug port on AppNode[*AppNode*] in [*AppSpace*] in Domain[*Domain*], *Reason*	The debug port on the AppNode could not be enabled. The AppNode is not running or is not reachable. In this case, the following additional message is displayed: TIBCO-BW-ADMIN-500313: AppNode [*AppNode*] is not running or could not be contacted. Make sure the AppNode is running and try again.
The debug port on the AppNode could not be disabled.	TIBCO-BW-ADMIN-CLI-500317: Failed to disable debugger on AppNode[*AppNode*] in [*AppSpace*] in Domain[*Domain*], *Reason*	The debug port on the AppNode could not be disabled. The AppNode is not running or is not reachable. Make sure the AppNode is running and try again.

Troubleshooting Archive Issues

Some archive issues and possible resolutions are listed below. This list is not complete but provides examples of messages that might be returned.

For a complete list of error codes, see the *TIBCO ActiveMatrix BusinessWorks™ Error Codes* guide.

Issue	Message	Resolution
The specified archive could not be uploaded; it could not be located.	TIBCO-BW-ADMIN-CLI-500433: Failed to upload archive [*Archive*], <CausedBy> TIBCO-THOR-FRWK-CMN-500101: Ear file [*Archive*] is not found.	The specified archive could not be found. Check the path and the archive filename and issue the command again.
The specified archive has already been uploaded to the specified domain.	TIBCO-BW-ADMIN-CLI-500433: Failed to upload archive [*archive*], <CausedBy> TIBCO-BW-ADMIN-500447: Archive [*Archive*] is already present in the domain. Use -replace option to replace the existing archive.	The specified archive has already been uploaded to the specified domain. To replace the archive, upload the archive again, using the `-replace` option with the `upload` command. Use the `show archives` command to view archives, versions, time uploaded, size, and path.
The specified archive could not be uploaded to all machines in the domain.	TIBCO-BW-ADMIN-CLI-500432: Failed to upload ear file [*Archive*] to some machines in the domain.	Not all machines that the domain has been expanded to are reachable. Check that machines are running.
The specified archive has already been uploaded. The application archive has already uploaded and deployed.	TIBCO-BW-ADMIN-CLI-500438: The application [*Application*] from archive [*Archive*] has been deployed to these AppSpaces: *AppSpace*	You already uploaded the archive and deployed the specified application from the archive. The message displays the AppSpaces the application has been deployed to. Use the `show applications` command to view applications, versions, and statuses.
The specified archive could not	TIBCO-BW-ADMIN-CLI-500434: Failed to delete archive [*Archive*]	The specified archive may not exist in the current domain

Issue	Message	Resolution
be deleted.	<CausedBy> *Reason*	context. If this is the case, the following message is appended: TIBCO-BW-ADMIN-500418: Application archive file [*Archive*] not found in the domain [*Domain*]. Use the `show archives` command to view archives, versions, time uploaded, size, and path. The specified archive may be deployed and cannot be deleted. In this case, the following message is displayed: TIBCO-BW-ADMIN-500450: Archive [*Archive*] has been deployed to the AppSpaces: [*AppSpaces*].
The specified application is not in sync with the specified archive.	TIBCO-BW-ADMIN-CLI-500439: Applications are out of sync with the archive they were deployed from, They have to be redeployed to keep them in sync.	The application is no longer in sync with the archive. Use the `deploy -replace` command to replace the existing version of the application with the version from the archive. This message is displayed with other messages, for example, if the archive could not be uploaded.

Troubleshooting Application Issues

Some application issues and possible resolutions are listed below. This list is not complete but provides examples of messages that might be returned.

For a complete list of error codes, see the *TIBCO ActiveMatrix BusinessWorks™ Error Codes* guide.

Issue	Message	Resolution
The specified application could not be deployed; no AppNodes exist.	TIBCO-BW-ADMIN-CLI-300432: Deployed application [*Application:Version*], The AppSpace [*AppSpace*] does not have any AppNodes.	No AppNodes have been created in the specified AppSpace. AppNodes are required for deployment. Create one or more AppNodes and issue the `deploy` command again.
The specified application could not be deployed. An application has already been deployed for that archive.	TIBCO-BW-ADMIN-CLI-300433: An application is already deployed with the archive [*Archive*].	The specified application has already been deployed for the specified archive. Use the `show applications` command to view it.
The specified application version format is not supported.	TIBCO-BW-ADMIN-CLI-500407: Version [*Version*] is not valid. Only <major>.<minor> version format is supported.].	An application version must be specified as a major.minor version within an AppSpace. Check the formatting of the version number.
The specified application could not be started as it is not found in the domain.	TIBCO-BW-ADMIN-CLI-500409: Failed to start application [*Application:Version*]. <CausedBy> TIBCO-BW-ADMIN-500401: Application [*Application*] not found in the Domain [*Domain*]	The specified application is not found in the specified domain. Use the `show applications` command to verify the application.

Issue	Message	Resolution
The specified application could not be started.	TIBCO-BW-ADMIN-CLI-500409: Failed to start application [*Application:Version*]. <CausedBy> TIBCO-BW-ADMIN-500444: Failed to start Application in AppNode [*AppNode*]. Check the AppNode log files for messages starting with TIBCO-THOR-FRWK, TIBCO-BW-FRWK, or TIBCO-BW-SR-FRWK for details. Application State [Start failed], reason: [*Reason*]	The specified application failed to start. This could be caused by unresolved shared resources, missing constraints, or missing components. If you see this message, open the log file for the AppNode (in the *BW_HOME*`\domains\<domain>\<AppSpace>\<AppNode>\log` folder) and check for messages starting with: • TIBCO-THOR-FRWK • TIBCO-BW-FRWK • TIBCO-BW-SR-FRWK These messages should help you identify the source of the issue. You may need to adjust the logging level for the log file. See AppNode Logging for information. An application might not start if the specified AppNode is not running or is not reachable. In this case, the following message is displayed: TIBCO-BW-ADMIN-CLI-500409: Failed to start application [*Application:Version*]. <CausedBy> TIBCO-BW-ADMIN-500313: AppNode [*AppNode*] is not running or cannot be contacted.
The specified application has been deployed but could not be started.	TIBCO-BW-ADMIN-C*Reason*LI-500414: Deployed application from the archive [*Archive*], however not all application instances started. <CausedBy> TIBCO-BW-ADMIN-500313: AppNode [*AppNode*] is not running or cannot	The application instance cannot be started on the specified node, as it is not running or is unreachable. Check that the BWAgent the AppNode is registered to is reachable. Check that the AppNode is running.

Issue	Message	Resolution
	be contacted.	
The specified application is not in sync with the archive.	TIBCO-BW-ADMIN-CLI-500439: Applications are out of sync with the archive that they were deployed from. They have to be redeployed to keep them in sync.	The EAR file has been replaced with a new version or a new file of the same version. Redeploy the application using the -replace option: `deploy -replace <EAR>`.

Troubleshooting Admin UI Issues

Some Admin UI issues and possible resolutions are listed below. This list is not complete but provides examples of messages that might be returned.

For a complete list of error codes, see the *TIBCO ActiveMatrix BusinessWorks™ Error Codes* guide. When a new version of the product is installed, you may need to open the Admin UI **Agents** page and reconnect to the BWAgent.

Issue	Message	Resolution
The specified BWAgent could not be registered in the Admin UI.	Failure registering agent with [*URL*].	This error can be caused by several issues: • The specified BWAgent at the URL provided in the Register Agent dialog could not be located. Verify the URL and register the BWAgent again. • A BWAgent with the specified name exists. Verify the name of the BWAgent and register the BWAgent again. • The specified BWAgent might not be running. Make sure that the BWAgent is running and register the BWAgent again.
The specified	The status of	The specified BWAgent might not be running. Make

Issue	Message	Resolution
BWAgent is unreachable.	`Unreachable` is displayed on the **Agent Management** page for the specified BWAgent.	sure that the BWAgent is running and register the BWAgent again.
The details for a selected runtime entity cannot be displayed.	Error running operation 'getStates', status '0', reason 'Connection refused: no further information'.	Since the Admin UI page for the selected runtime entity was displayed, either the runtime entity was deleted or the BWAgent was stopped. Use BWAdmin commands for the registered BWAgent to verify the state of runtime entities and check the status of the BWAgent.
The specified domain exists.	TIBCO-BW-TEAAGENT-500309: Failed to created Domain [*Domain*] TIBCO-BW-ADMIN-500101: Domain [*Domain*] exists, check this and retry.	The specified domain exists. Domain names must be unique; enter a different name. View existing domains on the **Domain Management** page.
The specified domain could not be created. The BWAgent on the remote machine is not running.	TIBCO-BW-TEAAGENT-500309: Failed to created Domain [*Domain*] TIBCO-BW-ADMIN-500004: Error invoking [joinmachine] method on the agent [*bwagent*], The BW Agent [*bwagent*] on the remote machine is not running.	Verify the BWAgent name. Check that the BWAgent on the remote machine is running.
The specified AppSpace in	TIBCO-BW-TEAAGENT-500425: AppSpace	The specified AppSpace exists. AppSpace names must be unique; enter a different name. View

Issue	Message	Resolution
the selected domain exists.	[*AppSpace*] exists in domain [*Domain*].	existing AppSpaces on the **AppSpaces** page.
The AppSpace status is `Degraded` and cannot be started.	The status of `Degraded` is displayed on the **AppSpaces** page for the specified AppSpace.	An AppSpace can only be started if it contains at least one AppNode. Check the `minNodes` value on the **AppSpaces** page and then pivot to the **AppNodes** page. Verify that the minimum number of AppNodes has been created. When the `minNodes` value is reached, the status of the AppSpace is changed to `Stopped`.
The specified AppNode in the selected AppSpace and domain exists.	TIBCO-BW-TEAAGENT-500313: Failed to create AppNode [*AppNode*] in AppSpace [*AppSpace*] in Domain [*Domain*], TIBCO-BW-ADMIN-500301: The AppNode [*AppNode*] exists in the AppSpace [*AppSpace*] Domain [*Domain*].	The specified AppNode exists. AppNode names must be unique; enter a different name. View existing AppNodes on the **AppNodes** page.
The specified archive could not be uploaded.	Failed to Upload. TIBCO-BW-ADMIN-500447: Archive [*Archive*] is already present in the domain. Use -replace option to replace the existing archive.	The specified archive has already been uploaded to the specified domain. To replace the archive, select the **Replace any version** checkbox in the Upload EAR File dialog, and upload the archive again.
The specified archive could not be deployed.	TIBCO-BW-TEAAGENT-300016: Deployed Application [*Application*] in	The specified AppSpace does not have any associated AppNodes. Click **Create AppNodes** on the **AppNodes** page. Select the specified AppSpace in the **Create AppNode** dialog.

Issue	Message	Resolution
	AppSpace [*AppSpace*] of Domain [*Domain*]. The AppSpace [*AppSpace*] does not have any AppNodes.	
The specified application could not be started.	TIBCO-BW-TEAAGENT-500315: Failed to deploy Application from archive [*Archive*] TIBCO-BW-ADMIN-500444: Failed to start Application in AppNode [*AppNode*]. Check the AppNode log files for messages starting with TIBCO-THOR-FRWK, TIBCO-BW-FRWK, or TIBCO-BW-SR-FRWK for details. Application State [Start failed], reason: *Reason*	The specified application failed to start. This could be caused by unresolved shared resources, missing constraints, or missing components. If you see this message, open the log file for the AppNode (in the *BW_HOME*`\domains\<domain>\<AppSpace>\<AppNode>\log` folder) and check for messages starting with: • TIBCO-THOR-FRWK • TIBCO-BW-FRWK • TIBCO-BW-SR-FRWK These messages should help you identify the source of the issue. You may need to adjust the logging level for the log file. For more information, see AppNode Logging.
The specified application could not be started.	TIBCO-BW-TEAAGENT-500410: Failed to start Application TIBCO-BW-ADMIN-500313: AppNode [*AppNode*] is not running or cannot be contacted.	The specified AppSpace may not have associated AppNodes or it might be on a BWAgent that is not reachable. An application can only be started if the specified AppNode is running. Pivot to the AppNodes page and verify the status of AppNodes in the AppSpace. If no AppNodes exist, create at least one and try to start the application again. If the minimum number of AppNodes is specified when the AppSpace is created, that number of AppNodes needs to exist. If the application is on a remote machine, the machine might be down or the BWAgent might not

Issue	Message	Resolution
		be running.
The specified archive could not be deleted.	Archive could not be deleted. TIBCO-BW-ADMIN-500450: Archive [*Archive*] has been deployed to the AppSpaces: [*AppSpaces*].	The specified archive has been deployed. Undeploy the archive from the **Application Archives** page, then click **Delete** to delete the archive.
The application is not in sync with the archive.	The `Out of Sync` state is displayed for the application **Deployment state** on the **Applications** page.	Click the **Details** link to view the reason for the state. The `Out of Sync` state is displayed when the EAR file is replaced with a new version or a new file of the same version. The software can detect that the application for this archive was already deployed. Redeploy the application to resolve the `Out of Sync` state. The following error message can also be displayed for an out of sync situation: TIBCO-BWTEAAGENT-300015: Uploaded Archive [*Archive*] in Domain [*Domain*]. Application deployed in the following AppSpace(s) [*AppSpace*] is out of sync. Please redeploy.

Logging

Log files are generated for BWAdmin, BWAgent, AppNodes, and applications. Log files capture all executed commands, and depending on the logging level, the corresponding activities.

Log file configuration follows the Logback standard. (For more information about configuration parameters, see the Logback Project at `http://logback.qos.ch/`)

The log files created by BWAdmin and BWAgent are written to the *BW_HOME*`/logs` folder. Log files created by other utilities, such as `bwdesign`, are also written to this folder. AppNode log files are written to the `/log` folder for the AppNode. If you contact TIBCO Support, your support representative most likely asks you to send the appropriate log file.

Logging configurations are customized in the following files:

- For BWAdmin: *BW_HOME*`/bin/bwadmin-logback.xml`
- For BWAgent: *BW_HOME*`/bin/bwagent-logback.xml`
- For an AppNode: *BW_HOME*`/domains/domain/<AppSpace>/<AppNode>/logback.xml` (created after the AppNode has been started)

Logs can be output to the BWAdmin console and a log file. Log files can be retrieved by BWAdmin and displayed in the console.

To upload or download a Logback file, click the **Upload** or **Download** link, from the Admin UI.

Logging Levels

The global logging level for each type of log is set to INFO by default. Five levels are supported. Specifying a level includes all higher levels. Levels are listed below in order from lowest (least restrictive) to highest (most restrictive):

- TRACE: Records fine-grained informational events.
- DEBUG: Records fine-grained informational events that help in debugging. Useful for diagnostics.
- INFO: Records informational messages that highlight the progress of the application.

Useful for production mode.

- WARN: Records potentially harmful situations.
- ERROR: Records error events that are harmful enough to prevent the application from running.

Note: Setting the logging level to DEBUG can adversely affect the performance, especially when logging SOAP messages with attachments or mail with attachments. In such cases, we recommend fine-tuning the loggers to log at ERROR level instead of DEBUG.

For more information on log file components and error code ranges, see Debugging.

Log File Encoding

Log files are generated according to the configuration specified in the `logback.xml` file. If the encoding is not specified in the `logback.xml` file, the system default encoding is used when generating the log files.

To save the log files in a specific encoding like UTF-8, add the charset element to the File Appender Encoder configuration in the `logback.xml` file.

```
<appender name="FILE" class="ch.qos.logback.core.FileAppender">
    <file>test.log</file>
      <append>true</append>
        <encoder>
          <charset>UTF-8</charset>
            <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} -
%msg%n</pattern>
        </encoder>
  </appender>
```

Application Logging

You can generate separate log files for an application either by configuring the **Log** activity in TIBCO Business Studio for BusinessWorks or by modifying the `logback.xml` file of the AppNode.

To support application logging, the following prerequisites must be satisfied.

Before you begin

- A sifting appender `<appender-ref ref="APPLICATION-FILE"/>` must be present in the `logback.xml` file of the AppNode.
- To support Debugging a Specific Application on the AppNode, the logger BWApp must be present in the `logback.xml` file of the AppNode.

```
<!-- Do not modify this logger-->
<logger name="BWApp">
       <level value="ERROR"/>
 </logger>
```

- To avoid having hash (#) as part of the logger names in the log files, the appenders present in the `logback.xml` file must use the following encoder:
  - For AppNode:

```
<encoder
class="ch.qos.logback.core.encoder.LayoutWrappingEncoder">
          <layout
class="com.tibco.bw.extensions.logback.BWLoggerPatternLayou
t"/>
</encoder>
```

  - For TIBCO Business Studio for BusinessWorks:

```
<encoder
class="ch.qos.logback.core.encoder.LayoutWrappingEncoder">
          <layout
class="com.tibco.bw.extensions.logback.BWLoggerPatternLayoutSt
udio"/>
</encoder>
```

By default, new log files are created under the `{BW.HOME}/bw/6.x/logs` directory for TIBCO Business Studio for BusinessWorks, and for the AppNode, new log files are created under the `{APPNODE.HOME}/log` directory.

Note: You can specify a custom location for the creation of the log file. However you must use the variable `${fileName}` to define the name of the file.

Controlling the Output of the Log Activity

You can configure **Log** activity messages to be logged into separate files in one of the following ways:

- **By configuring the Log Activity in TIBCO Business Studio for BusinessWorks**

  You can configure the **Log** activity in TIBCO Business Studio for BusinessWorks to separate logs by application, process, or event type, depending on the options selected in the **Log** activity.

  For more information about options in the **Log** activity, see the "Log" section in the *TIBCO ActiveMatrix BusinessWorks™ Binding and Palettes Reference*.

- **By configuring the Logger in the AppNode's `logback.xml` file**

  You can separate the logs from the **Log** activity based on your application, without changing the **Log** activity just by modifying the `logback.xml` file of the AppNode.

  - For one application, add a logger in the `logback.xml` file of the AppNode as follows:

    ```
    <logger
    name="BWApp.#APPNAME#.com.tibco.bw.palette.generalactivities.L
    og"  additivity="false">
        <level value="DEBUG"/>
        <appender-ref ref="APPLICATION-FILE"/>
     </logger>
    ```

    APPNAME is the name of the application.

  - For multiple or all applications, add the previous logger for each application or follow the steps specified in the Creating Separate Log Files for Each Application on the AppNode section.

Displaying Log Activity Messages on TIBCO Business Studio for BusinessWorks Console

To display the **Log** activity messages in the TIBCO Business Studio for BusinessWorks console, the additivity attribute can be omitted from the logger if the root logger is using STDOUT as its appender.

Note: By modifying the `logback.xml` file, the logs can only be separated by application and not by process or eventType.

Creating Separate Log Files for Each Application on the AppNode

There are two steps to generate separate log files for each application running on an AppNode:

1. Change the value of the property `bw.engine.separate.logs.by.app` to true from the default value false in the `config.ini` file of the AppSpace or the AppNode.

   To enable this property through TIBCO Business Studio for BusinessWorks, pass it as a VM argument using the `-D` option.

   > **Note:** If the `bw.engine.separate.logs.by.app` property is set at the AppNode level, this setting takes precedence over the property set at the AppSpace level. Restart the AppNode when the property is updated.

2. Add or modify the loggers in the `logback.xml` file of AppNode to use `<appender-ref ref="APPLICATION-FILE" />`.

> **Note:** If there are any logs that are not specific to an application, and logs generated from all loggers other than supported loggers, are written to the defaulf `bwappnode.log` file.

To create separate log files for all applications in the AppNode, without modifying the **Log** activity and without adding multiple loggers to the `logback.xml` file, set the property `bw.engine.separate.logs.by.app` to `true` and if not already present add the `<appender-ref ref="APPLICATION-FILE" />` to the logger `com.tibco.bw.generalactivities.palette` as follows:

```
<logger name="com.tibco.bw.palette.generalactivities.Log"
additivity="false">
    <level value="DEBUG"/>
     <appender-ref ref="APPLICATION-FILE"/>
 </logger>
```

The following examples demonstrate some common use cases:

1. To separate supported logs by application name, modify the root logger's appender to use the new sifting appender.

```
 <root level="ERROR">
    <appender-ref ref="APPLICATION-FILE" />
</root>
```

This configuration causes all ERROR logs from all the supported loggers to be separated by application name. The level of individual loggers can be set to a desired value such as DEBUG, ERROR, or INFO.

2. To separate logs of only `com.tibco.bw.core` by application name, modify the logger as follows:

```
<logger name="com.tibco.bw.core" additivity="false">
<level value="ERROR"/>
<appender-ref ref="APPLICATION-FILE"/>
</logger>
```

3. To separate all palette logs by application name, modify the `com.tibco.bw.palette` logger as follows:

```
<logger name="com.tibco.bw.palette" additivity="false">
<level value="DEBUG"/>
<appender-ref ref="APPLICATION-FILE"/>
</logger>
```

This configuration causes all DEBUG logs from all the palettes in the AppNode to be separated by application name.

A similar configuration can be extended to `com.tibco.bw.sharedresource` and `com.tibco.bx`.

```
<logger name="com.tibco.bw.sharedresource" additivity="false">
<level value="DEBUG"/>
<appender-ref ref="APPLICATION-FILE"/>
</logger>
<!-- ---For bx--->
<logger name="com.tibco.bx" additivity="false">
<level value="DEBUG"/>
<appender-ref ref="APPLICATION-FILE"/>
</logger>
```

4. To separate logs of only JMS Connection shared resource by application name, a new logger must be added to the `logback.xml` file.

```
<logger name="com.tibco.bw.sharedresource.jms" additivity="false">
<level value="DEBUG"/>
<appender-ref ref="APPLICATION-FILE"/>
</logger>
```

This configuration causes all DEBUG logs from all JMS Connection shared resources in the AppNode to be separated by application name. All other shared resources follow the behavior of the `com.tibco.bw.sharedresource` logger.

For TIBCO Business Studio for BusinessWorks, the `bwappnode.log` file is created in the same location where the application logs are generated.

If the `additivity` attribute is not set to `false`, the logger continues to use the appenders of the parent logger all the way up to the root logger until it finds a logger whose additivity is set to false.

Caution: If not configured correctly, this can lead to duplicate logging.

Debugging a Specific Application on the AppNode

- To enable DEBUG logging on a specific application of the supported loggers, deployed on the AppNode, where multiple applications are running on the AppNode, you can add a new logger to the existing `logback.xml` file of the AppNode.

  In this scenario, the property `bw.engine.separate.logs.by.app` need not be set to true.

  ```
  <logger name="BWApp.#APPNAME#">
      <level value="DEBUG"/>
  </logger>
  ```

  Where 'APPNAME' is the name of the application whose debug logs are desired.

  For example, if there are three applications running on the AppNode, App1.application, App2.application, and App3.application, the debug logs can be turned on only for App2.application by adding a logger `BWApp.#App2.application#` to the `logback.xml` file.

```
<logger name="BWApp.#App2.application#">
    <level value="DEBUG"/>
</logger>
```

The new logger can be appended by any of the supported loggers.

- If you want debug logs only for the **HTTP** palette of the App2.application, add a new logger to the `logback.xml` as follows:

```
<logger name="BWApp.#App2.application#.com.tibco.bw.palette.http">
    <level value="DEBUG"/>
</logger>
```

- Use `<appender-ref ref="APPLICATION-FILE"/>` to separate log files for each supported logger.

```
<logger name="BWApp.#APPNAME#" additivity="false">
      <level value="DEBUG"/>
      <appender-ref ref="APPLICATION-FILE"/>
</logger>
```

Note: Supported log level value for these appenders is DEBUG and TRACE only.

Supported Loggers

You can generate separate logs for each application for the following loggers only:

- `com.tibco.bw.core`
- `com.tibco.bw.palette` and the hierarchical children. For example, `com.tibco.bw.palette.http`, `com.tibco.bw.palette.file`
- `com.tibco.bw.sharedresource` and the hierarchical children. For example, `com.tibco.bw.sharedresource.jdbc`, `com.tibco.bw.sharedresource.jms`
- `com.tibco.bx`

Backward Compatibility for Application Logging

You can run applications created in the versions prior to the ActiveMatrix BusinessWorks 6.5.0 version having **Log** activities, in the new version with the new or old `logback.xml` file. There is no change in the behavior of an application, and it continues to work as is.

If the property `bw.engine.separate.logs.by.app` is set to `false`, there is no change in the behavior of an application, and it continues to work as is.

AppNode Logging

AppNode logging is enabled for every AppNode.

The AppNode log file is named `bwappnode.log`. It is written to the *BW_HOME*`/domains/<domain>/<AppSpace>/<AppNode>/log` folder.

An AppNode log file is created when an AppNode is started. By default, AppNode logs are written to a file appender only. The logs can be viewed in a text editor or displayed in the BWAdmin console. The default logging level is ERROR.

In addition to AppNode logging, execution statistics are collected through Logback. For information, see Integrating Process Statistics Collection Using Logback.

To view and change the logging level for a single AppNode:

Procedure

1. Create and start an AppNode.
2. Open the `logback.xml` file in the AppNode root folder in a text editor: *BW_HOME*`/domains/<domain>/<AppSpace>/<AppNode>`
3. Change the ROOT level at the end of the file as needed.
4. Save the `logback.xml` file.

   By default, the new configuration is reloaded in 60 seconds. There is no need to restart the AppNode for the logging level to take effect.

   To change the scan period, add the `scanPeriod` attribute to the `configuration` element in the `logback.xml` file. For example:

```
<configuration scan="true" scanPeriod="10 seconds">
```

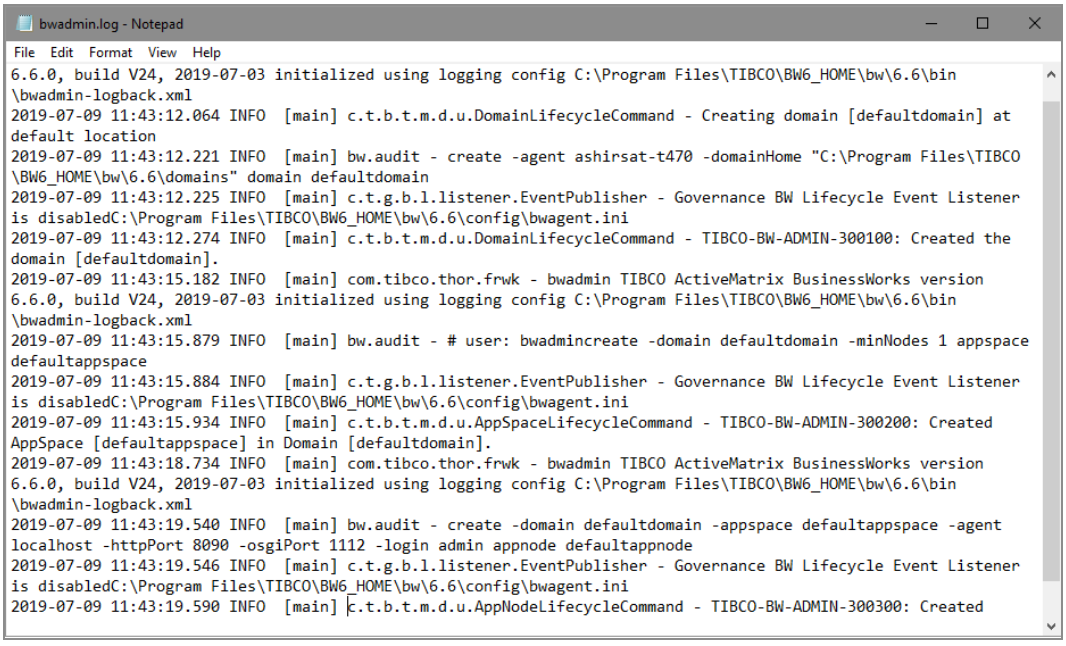
5. Save the file, start the AppNode, and the application.
6. Open the `bwappnode.log` file in the root folder of the AppNode to see what has been logged. The log file can be opened in the BWAdmin console with the following command from the containing AppSpace: `getlogfile appnode`*AppNode*

BWAdmin Logging

BWAdmin creates a log file called `bwadmin.log` that is written to the *BW_HOME*`/logs` folder. The default log is configured as a daily roller appender and is automatically compressed as a ZIP file. The default logging level is INFO. The Logback configuration file is *BW_HOME*`/bin/bwadmin-logback.xml`.

A BWAdmin log file is created on installation, showing the runtime entities that are created by default (domain, AppSpace, and AppNode). The contents of the default log file look similar to the following image.

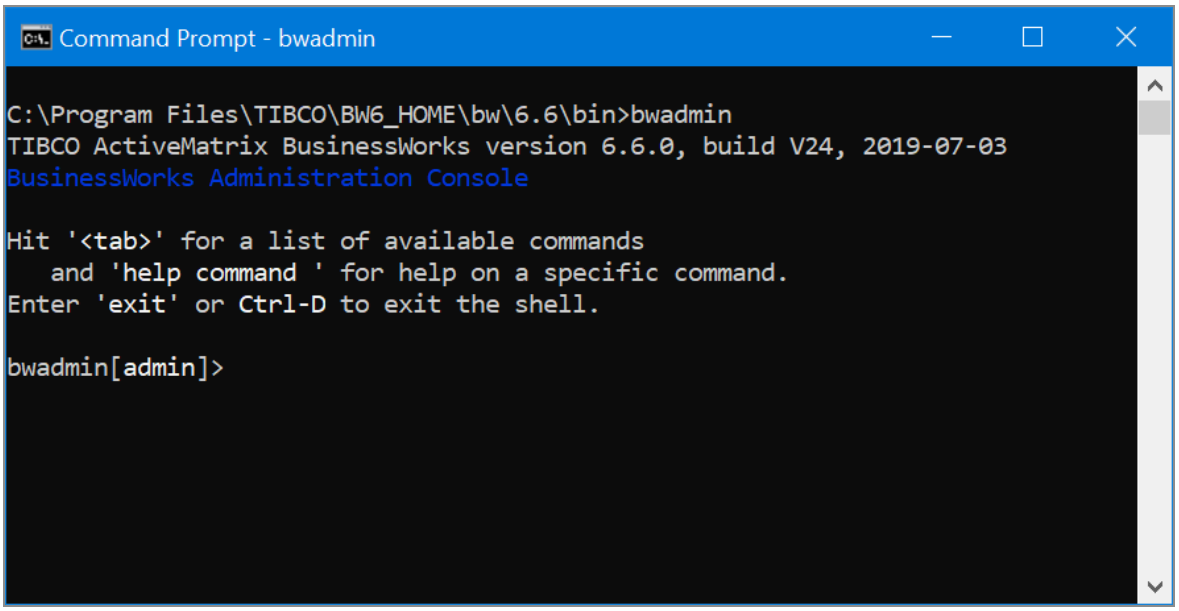
BWAdmin Log File on Install



To view and change the logging level for BWAdmin, follow these steps.

Procedure

1. Start BWAdmin. Notice that no INFO messages are displayed:

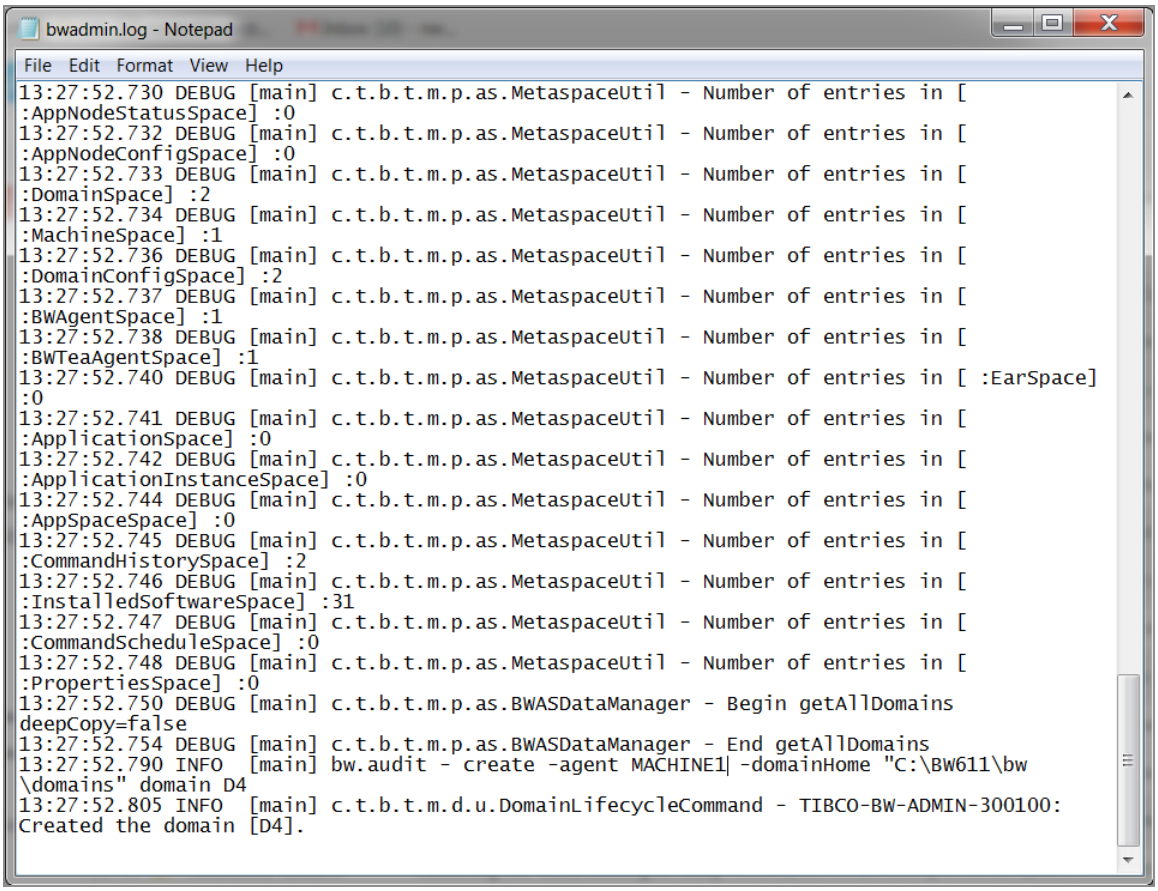


   If BWAdmin could not be started, informational messages are displayed to help you track down the cause.

2. Exit BWAdmin.
3. Open *BW_HOME*`/bin/bwadmin-logback.xml` in a text editor. Change the ROOT level setting (indicated below in bold font) at the end of the file as needed. This changes the level for the file appender.

```
<root level="INFO">
    <appender-ref ref="STDOUT" />
    <appender-ref ref="FILE" />
```

4. Save the Logback configuration file and start BWAdmin. The configuration file is reloaded in 30 seconds by default. Create a domain.
5. Open the log file at *BW_HOME*`/logs/bwadmin.log`. More detail is captured in the log file.



Admin Message Codes

This section describes the informative messages that are generated by the entities associated with the ActiveMatrix BusinessWorks Administrator.

Message Codes	Message
TIBCO-BW-ADMIN-100001	{0}
TIBCO-BW-ADMIN-200001	{0}
TIBCO-BW-ADMIN-300010	Registered TEA Agent [{0}] with TEA Server [{1}].
TIBCO-BW-ADMIN-300100	Created the domain [{0}].

Message Codes	Message
TIBCO-BW-ADMIN-300101	Added Machine [{0}] to Domain [{1}]
TIBCO-BW-ADMIN-300102	Removed Machine [{0}] from Domain [{1}]
TIBCO-BW-ADMIN-300103	Deleted the Domain [{0}].
TIBCO-BW-ADMIN-300104	Registered BWAgent [{0}] to Domain [{1}]
TIBCO-BW-ADMIN-300105	Unregistered BWAgent [{0}] from AppSpace [{1}] in Domain [{2}]
TIBCO-BW-ADMIN-300200	Created AppSpace [{0}] in Domain [{1}].
TIBCO-BW-ADMIN-300201	Deleted AppSpace [{0}] in Domain [{1}].
TIBCO-BW-ADMIN-300202	Added Machine [{0}] to AppSpace [{1}] in Domain [{2}].
TIBCO-BW-ADMIN-300203	Removed AppSpace [{0}] from BWAgent [{1}].
TIBCO-BW-ADMIN-300204	Started AppSpace [{0}] in Domain [{1}].
TIBCO-BW-ADMIN-300205	Stopped AppSpace [{0}] in Domain [{1}].
TIBCO-BW-ADMIN-300206	Expanded AppSpace [{0}] to BWAgent [{1}] in Domain [{2}].
TIBCO-BW-ADMIN-300207	Deleted AppSpace [{0}] in Domain [{1}].
TIBCO-BW-ADMIN-300208	Updated configuration of AppSpace [{0}] in Domain [{1}].
TIBCO-BW-ADMIN-300300	Created AppNode [{0}] in AppSpace [{1}] in Domain [{2}].
TIBCO-BW-ADMIN-300301	Deleted AppNode [{0}] in AppSpace [{1}] in Domain [{2}].
TIBCO-BW-ADMIN-300302	Started AppNode [{0}] in AppSpace [{1}] in Domain [{2}].
TIBCO-BW-ADMIN-300303	Stopped AppNode [{0}] in Domain [{1}].

Message Codes	Message
TIBCO-BW-ADMIN-300304	Enabled console on AppNode [{0}] in AppSpace [{1}] and Domain [{2}].
TIBCO-BW-ADMIN-300305	Disabled console on AppNode [{0}] in AppSpace [{1}] and Domain [{2}].
TIBCO-BW-ADMIN-300306	Disabled debugger on AppNode [{0}] in AppSpace [{1}] in Domain [{2}]
TIBCO-BW-ADMIN-300307	Retrieved BWEngine information from AppNode [{0}] in AppSpace [{1}] and Domain [{2}].
TIBCO-BW-ADMIN-300400	Deployed Application [{0}:{1}]
TIBCO-BW-ADMIN-300401	Undeployed Application [{0}:{1}]
TIBCO-BW-ADMIN-300402	Updated Application[{0}:{1}]
TIBCO-BW-ADMIN-300403	Deployed Application [{0}:{1}]
TIBCO-BW-ADMIN-300404	Undeployed Application [{0}:{1}]
TIBCO-BW-ADMIN-300405	Started Application [{0}:{1}]
TIBCO-BW-ADMIN-300406	Stopped Application [{0}:{1}]
TIBCO-BW-ADMIN-300407	Hotfix to application [{0}:{1}] is deployed.
TIBCO-BW-ADMIN-300408	Hotfix to application [{0}:{1}] is undeployed.
TIBCO-BW-ADMIN-300409	Configured application [{0}:{1}].
TIBCO-BW-ADMIN-300410	Paused Application [{0}:{1}]
TIBCO-BW-ADMIN-300411	Resumed Application [{0}:{1}]
TIBCO-BW-ADMIN-300412	Application [{0}:{1}] is already deployed. Use the config

Message Codes	Message
	command to change the configuration.
TIBCO-BW-ADMIN-300413	Enabled statistics collection for Application [{0}:{1}].
TIBCO-BW-ADMIN-300414	Disabled statistics collection for Application [{0}:{1}].
TIBCO-BW-ADMIN-300500	Uploaded Archive [{0}]
TIBCO-BW-ADMIN-300501	Removed Archive [{0}]
TIBCO-BW-ADMIN-300510	Archive [{0}] was created with the BusinessStudio version: {1}
TIBCO-BW-ADMIN-300511	Debug port enabled on AppNode[{0}] in AppSpace[{1}] in Domain [{2}]
TIBCO-BW-ADMIN-400000	BW6_INFR_WARN_FORMAT = {0}
TIBCO-BW-ADMIN-400501	Name is longer than 100 characters. Truncating to meet length limit.
TIBCO-BW-ADMIN-400502	Name includes invalid characters that have been removed to comply with naming conventions.\nAllowed characters are upper and lower case characters of the alphabet as well as digits, " ", "." and "-".\nName has been changed to [{0}].
TIBCO-BW-ADMIN-500000	BW6_INFR_ERROR_FORMAT = {0}
TIBCO-BW-ADMIN-500001	Failed to perform operation [{0}] on [{1}] in the Domain [{2}] due to an error in initializing the data manager, <CausedBy> {3}
TIBCO-BW-ADMIN-500002	Error while checking if the entity [{0}] is present in the Domain [{1}], <CausedBy> {2}.
TIBCO-BW-ADMIN-500004	Error invoking [{0}] method on the agent [{1}], <CausedBy> {2}
TIBCO-BW-ADMIN-500005	Error while uploading the application archive file into the

Message Codes	Message
	data store, <CausedBy> {0}
TIBCO-BW-ADMIN-500006	Failed to initialize transport, <CausedBy> {0}
TIBCO-BW-ADMIN-500007	Unsupported value for bw.admin.mode [{0}]
TIBCO-BW-ADMIN-500008	Error in initializing data manager, <CausedBy> {0}
TIBCO-BW-ADMIN-500009	Cannot find or read file [{0}].
TIBCO-BW-ADMIN-500010	Invalid configuration content or format in file [{0}].
TIBCO-BW-ADMIN-500011	Remote command execution failed, <CausedBy> {0}
TIBCO-BW-ADMIN-500012	Configuration error in `bwagent.ini` file. Remote request timeout invalid.
TIBCO-BW-ADMIN-500013	Port [{0}] is already taken by AppNode [{1}] belonging to AppSpace [{2}] in the Domain [{3}]
TIBCO-BW-ADMIN-500014	Port [{0}] is in use already.
TIBCO-BW-ADMIN-500015	Conflicting HTTP and OSGi console ports. Port [{0}]
TIBCO-BW-ADMIN-500016	Configuration error in `bwagent.ini` file. Value given in `bw.agent.technology.as.remote.status.requestTimeout` is invalid.
TIBCO-BW-ADMIN-500017	BWAgent is running, Stop the remote agent, delete all the files under the domains folder, and retry.
TIBCO-BW-ADMIN-500100	Domain [{0}] not found, check this and retry.
TIBCO-BW-ADMIN-500101	Domain [{0}] exists, check this and retry.
TIBCO-BW-ADMIN-500102	Failed to create domain [{0}]

Message Codes	Message
TIBCO-BW-ADMIN-500103	Domain home folder [{0}] does not exist. Verify if the folder exists and use the '/' character in the folder path.
TIBCO-BW-ADMIN-500104	Machine [{0}] is not present in the Domain [{1}].
TIBCO-BW-ADMIN-500105	Machine [{0}] could not be added to the Domain [{1}], {2}.
TIBCO-BW-ADMIN-500106	Failed to delete Domain [{0}], {1}.
TIBCO-BW-ADMIN-500107	Machine [{0}] could not be removed from the Domain [{1}], {2}.
TIBCO-BW-ADMIN-500108	Machine [{0}] is already part of Domain [{1}]
TIBCO-BW-ADMIN-500109	The Domain [{0}] has AppSpaces associated with it. Use the `–force` option to override.
TIBCO-BW-ADMIN-500108	Agent [{0}] is already part of domain [{1}]
TIBCO-BW-ADMIN-500109	Domain [{1}] could not be expanded to the Agent [{0}], {2}.
TIBCO-BW-ADMIN-500110	Domain [{1}] could not be removed from the Agent [{0}], {2}.
TIBCO-BW-ADMIN-500111	Agent [{0}] is not part of the Domain [{1}].
TIBCO-BW-ADMIN-500112	Domain [{0}] not found, check this and retry.
TIBCO-BW-ADMIN-500113	Domain [{0}] is not a local domain.
TIBCO-BW-ADMIN-500201	AppSpace [{0}] in Domain [{1}] not found.
TIBCO-BW-ADMIN-500202	The AppSpace [{0}] is already present in the Domain [{1}].
TIBCO-BW-ADMIN-500204	AppSpace [{0}] exists with agent [{1}].
TIBCO-BW-ADMIN-500205	AppSpace [{0}] in Domain [{1}] could not be deleted, {2}.

Message Codes	Message
TIBCO-BW-ADMIN-500206	Failed to register Agent [{0}] to AppSpace [{1}], {2}.
TIBCO-BW-ADMIN-500207	Failed to unregister Agent [{0}] from AppSpace [{1}], {2}.
TIBCO-BW-ADMIN-500208	Adding Machine to an AppSpace is not supported in 'local' admin mode.
TIBCO-BW-ADMIN-500209	Removing the Machine from an AppSpace is not supported in 'local' admin mode.
TIBCO-BW-ADMIN-500210	AppSpace [{0}] in Domain [{1}] did not start completely, {2}.
TIBCO-BW-ADMIN-500211	AppSpace [{0}] in Domain [{1}] could not be stopped, {2}.
TIBCO-BW-ADMIN-500212	AppSpace [{0}] in Domain [{1}] could not be deleted, {2}.
TIBCO-BW-ADMIN-500213	AppSpace [{0}] in Domain [{1}] does not have any AppNodes.
TIBCO-BW-ADMIN-500214	Failed to deploy the {0} [{1}] from URL {2} into AppSpace {3}
TIBCO-BW-ADMIN-500215	AppSpace handler [{0}] not found
TIBCO-BW-ADMIN-500216	AppSpace [{0}] has AppNodes associated with it. Delete the AppNodes first and retry or use the `-force` option to override
TIBCO-BW-ADMIN-500217	AppSpace [{0}] has Applications or Libraries associated with it. Undeploy them first and retry or use the `-force` option to override
TIBCO-BW-ADMIN-500218	AppSpace [{0}] is not expanded to BWAgent [{1}].
TIBCO-BW-ADMIN-500219	AppSpace [{0}] is not scaled across multiple installations. Use the `delete` command to delete it.
TIBCO-BW-ADMIN-500220	AppSpace [{0}] is scaled across multiple BW Agents. Cannot be deleted. Use the `-force` option to override.

Message Codes	Message
TIBCO-BW-ADMIN-500221	AppSpace [{0}] could not be updated, {1}
TIBCO-BW-ADMIN-500222	Configuration of AppSpace [{0}] in Domain [{1}] failed, {2}
TIBCO-BW-ADMIN-500223	Configuration failed on one or more remote agents.
TIBCO-BW-ADMIN-500224	AppSpace [{0}] has already reached its limit of applications [{1}]
TIBCO-BW-ADMIN-500225	Only one version of Application [{0}] can be deployed in an AppSpace at the same time.
TIBCO-BW-ADMIN-500226	The registration of BWAgent to the domain [{0}] failed. Cannot create AppNodes in the AppSpace [{1}] on the remote BWAgent [{2}]
TIBCO-BW-ADMIN-500227	The registration of BWAgent to the AppSpace [{0}] failed. Cannot create AppNodes in the AppSpace [{0}] on the remote BWAgent [{1}]
TIBCO-BW-ADMIN-500228	Invalid value supplied as minNodes. minNodes must be an Integer greater than 0.
TIBCO-BW-ADMIN-500229	AppSpace [{0}] is expanded to the BWAgent [{1}] on the file system, However the agent failed to update the status in the datastore. <CausedBy> {2}
TIBCO-BW-ADMIN-500230	AppSpace [{0}] is expanded to the BWAgent [{1}] on the file system, However one or multiple applications failed to deploy into this AppSpace on the remote machine, check the status of the applications in this AppSpace for details. <CausedBy> {2}
TIBCO-BW-ADMIN-500231	AppSpace [{0}] is expanded to the BWAgent [{1}] on the file system, However one or multiple applications failed to deploy into this AppSpace on the remote machine, check the status of the applications in this AppSpace for details.

Message Codes	Message
TIBCO-BW-ADMIN-500232	Errors while deleting AppNodes and/or AppSpace instances on local and/or remote machines.
TIBCO-BW-ADMIN-500233	Cannot find AppSpace specific TRA file at [{0}]
TIBCO-BW-ADMIN-500234	Failed to obtain the AppSpace TRA file from [{0}], <CausedBy> [{1}]
TIBCO-BW-ADMIN-500235	Failed to write AppSpace TRA file [{0}], <CausedBy> [{1}]
TIBCO-BW-ADMIN-500300	The AppNode [{0}] does not exist in AppSpace [{1}] and Domain [{2}].
TIBCO-BW-ADMIN-500301	The AppNode [{0}] exists in the AppSpace [{1}] Domain [{2}].
TIBCO-BW-ADMIN-500302	Failed to create AppNode [{0}] in AppSpace [{1}] in Domain [{2}], {3}
TIBCO-BW-ADMIN-500303	Start of AppNode [{0}] in AppSpace [{1}] in Domain [{2}] timed out.
TIBCO-BW-ADMIN-500304	AppNode [{0}] in AppSpace [{1}] in Domain [{2}] did not start, <CausedBy> {3}
TIBCO-BW-ADMIN-500305	AppNode [{0}] in Domain [{1}] did not stop, <CausedBy> {2}
TIBCO-BW-ADMIN-500306	Failed to delete AppNode [{0}] in AppSpace [{1}] in Domain [{2}], <CausedBy> {3}
TIBCO-BW-ADMIN-500307	Stop AppNode [{0}] in Domain [{1}] timed out.
TIBCO-BW-ADMIN-500308	Reached wrong AppNode [{0}] in AppSpace [{1}] in Domain [{2}]. Expected was [{3}].
TIBCO-BW-ADMIN-500309	Failed to enable console on AppNode [{0}] in AppSpace [{1}] in Domain [{2}], <CausedBy> {3}

Message Codes	Message
TIBCO-BW-ADMIN-500310	Failed to disable console on AppNode [{0}] in AppSpace [{1}] in Domain [{2}], <CausedBy> {3}
TIBCO-BW-ADMIN-500311	Failed to update AppNode [{0}] in data store, <CausedBy> {1}
TIBCO-BW-ADMIN-500312	Failed to configure AppNode [{0}], <CausedBy> {1}
TIBCO-BW-ADMIN-500313	AppNode [{0}] is not running or cannot be contacted
TIBCO-BW-ADMIN-500314	The AppNode [{0}] is still in the [{1}] state. Please stop the AppNode first or use the -force option.
TIBCO-BW-ADMIN-500315	No AppNodes are present in the AppSpace [{0}] in the Domain [{0}].
TIBCO-BW-ADMIN-500316	AppNode [{0}] log file does not exist or could not be read
TIBCO-BW-ADMIN-500317	Agent log file does not exist or could not be read
TIBCO-BW-ADMIN-500318	The AppNode [{0}] is running. Stop the AppNode first and try again.
TIBCO-BW-ADMIN-500319	Port is taken, <CausedBy> {0}
TIBCO-BW-ADMIN-500320	AppNode [{0}] encountered an Internal Server Error. Please check the log file of the AppNode for details.
TIBCO-BW-ADMIN-500321	Failed to disable debugger on AppNode [{0}] in AppSpace [{1}] in Domain [{2}]. <CausedBy> {3}
TIBCO-BW-ADMIN-500322	Failed to obtain BWEngine information on AppNode [{0}] in AppSpace [{1}] in Domain [{2}]. <CausedBy> {3}. Please check the log file of the AppNode for details.
TIBCO-BW-ADMIN-500323	Failed to enable port on the AppNode [{0}] in AppSpace [{1}] in Domain [{2}], <CausedBy> {3}

Message Codes	Message
TIBCO-BW-ADMIN-500323	Cannot find AppNode specific TRA file at [{0}]
TIBCO-BW-ADMIN-500324	Failed to obtain AppNode TRA file from [{0}], <CausedBy> [{1}]
TIBCO-BW-ADMIN-500325	Failed to write AppNode TRA file [{0}], <CausedBy> [{1}]
TIBCO-BW-ADMIN-500400	The supplied deployment artifact [{0}] does not exist.
TIBCO-BW-ADMIN-500401	Application [{0}] not found in the Domain [{1}] {2}
TIBCO-BW-ADMIN-500402	Failed to deploy application [{0}:{1}], <CausedBy> {2}
TIBCO-BW-ADMIN-500403	BW6_ADMIN_APP_UNDEPLOYMENT_FAILED = Failed to undeploy application [{0}:{1}], <CausedBy> {2}
TIBCO-BW-ADMIN-500404	The Application [{0}] is in the undeployed state.
TIBCO-BW-ADMIN-500405	The deployment artifact [{0}] supplied is not valid
TIBCO-BW-ADMIN-500406	Failed to deploy hotfix to application [{0}:{1}].
TIBCO-BW-ADMIN-500407	Version [{0}] is not valid. Only <major>.<minor> version format is supported.
TIBCO-BW-ADMIN-500408	Failed to undeploy hotfix to application [{0}:{1}].
TIBCO-BW-ADMIN-500409	Failed to start application [{0}:{1}], <CausedBy> {2}
TIBCO-BW-ADMIN-500410	Failed to stop application [{0}], <CausedBy> {1}
TIBCO-BW-ADMIN-500411	Deployment of [{0}] completed. The installation into running AppNodes failed for some AppNodes. Please check log files for more details.
TIBCO-BW-ADMIN-500412	Deployment of [{0}] completed. The start of the application or library in the running AppNodes failed for some AppNodes. Please check log files for more details.

Message Codes	Message
TIBCO-BW-ADMIN-500413	Failed to update the application [{0}] in the data store, <CausedBy> {1}
TIBCO-BW-ADMIN-500415	Failed to upload to one or multiple remote machines.
TIBCO-BW-ADMIN-500416	Error encountered installing {0} [{1}] [{2}] on AppNode [{3}]
TIBCO-BW-ADMIN-500417	Error encountered starting {0} on AppNode [{1}]
TIBCO-BW-ADMIN-500418	Application archive file [{0}] not found in the domain [{1}]
TIBCO-BW-ADMIN-500419	Application archive file [{0}] could not be deleted from the data store.
TIBCO-BW-ADMIN-500420	Name includes only invalid characters and does not comply with naming conventions. Allowed characters are upper and lower case characters of the alphabet as well as digits, '.' and '-'.
TIBCO-BW-ADMIN-500421	Configuration of Application [{0}] failed, <CausedBy> {1}
TIBCO-BW-ADMIN-500422	Failed to delete Application archive file on one or multiple remote machines.
TIBCO-BW-ADMIN-500423	BW6_ADMIN_APP_ALRDY_PRSNT =Application [{0}] exists in the domain [{1}] and it is deployed from the archive file [{2}]
TIBCO-BW-ADMIN-500424	Failed to read profile from [{0}]
TIBCO-BW-ADMIN-500425	Profile [{0}] is not present in the Archive [{1}]
TIBCO-BW-ADMIN-500426	Failed to export Profile [{0}] from the Archive {1}.
TIBCO-BW-ADMIN-500427	Failed to export configuration of Application [{0}] Version [{1}] AppSpace [{2}] Domain [{3}].
TIBCO-BW-ADMIN-500428	Failed to get configuration of Application [{0}] Version [{1}]

Message Codes	Message
	AppSpace [{2}] Domain [{3}] from the remote agent.
TIBCO-BW-ADMIN-500429	Failed to export configuration of Application Instance [{0}] Version [{1}] AppNode [{2}] AppSpace [{3}] Domain [{4}].
TIBCO-BW-ADMIN-500430	Application Instance [{0}] Version [{1}] on AppNode [{2}] AppSpace [{3}] Domain [{4}] is not found.
TIBCO-BW-ADMIN-500431	Error encountered pausing {0} [{1}] on AppNode [{2}]
TIBCO-BW-ADMIN-500432	Error encountered resuming {0} [{1}] on AppNode [{2}]
TIBCO-BW-ADMIN-500433	Pausing of [{0}] completed. Please check log files for more details.
TIBCO-BW-ADMIN-500434	Pausing of [{0}] completed. Please check log files for more details.
TIBCO-BW-ADMIN-500435	Failed to pause Application [{0}], <CausedBy> {1}
TIBCO-BW-ADMIN-500436	Failed to resume Application [{0}], <CausedBy> {1}
TIBCO-BW-ADMIN-500437	Failed to start in one or more AppNodes.
TIBCO-BW-ADMIN-500438	Failed to restore AppInstance configuration, <CausedBy> {0}
TIBCO-BW-ADMIN-500457	Error encountered starting process starters of {0} [{1}] [{2}] on AppNode [{3}]
TIBCO-BW-ADMIN-500439	Error encountered stopping process starters of {0} [{1}] [{2}] on AppNode [{3}]
TIBCO-BW-ADMIN-500440	Starting process starters of [{0}] completed. Please check log files for more details.
TIBCO-BW-ADMIN-500441	Stopping process starters of [{0}] completed. Please check log files for more details.

Message Codes	Message
TIBCO-BW-ADMIN-500442	Failed to start process starters of Application [{0}], <CausedBy> {1}
TIBCO-BW-ADMIN-500443	Failed to stop process starters of Application [{0}], <CausedBy> {1}
TIBCO-BW-ADMIN-500444	Failed to start Application in AppNode [{0}]. Check the AppNode log files for messages starting with TIBCO-THOR-FRWK, TIBCO-BW-FRWK, or TIBCO-BW-SR-FRWK for details. Application State [{1}], reason: {2}
TIBCO-BW-ADMIN-500445	AppSpace [{0}] is not found. Create the AppSpace first and retry.
TIBCO-BW-ADMIN-500446	The application [{0}] from archive [{1}] has been deployed to these AppSpaces: {2}
TIBCO-BW-ADMIN-500447	Archive [{0}] is already present in the domain. Use the –`replace` option to replace the existing archive.
TIBCO-BW-ADMIN-500448	Applications are out of sync with the archive that they were deployed from, They have to be redeployed to keep them in sync.
TIBCO-BW-ADMIN-500449	The archive [{0}] in the domain is replaced, Deploy the application again to apply the latest version.
TIBCO-BW-ADMIN-500450	Archive [{0}] has been deployed to the AppSpaces: {1}
TIBCO-BW-ADMIN-500451	Port is taken by AppNode[{0}] in AppSpace[{1}] in Domain[{2}]
TIBCO-BW-ADMIN-500452	Failed to obtain BWAgents of the machine [{0}].
TIBCO-BW-ADMIN-500453	HTTP interface and port of BWAgent [{0}] conflicted with the supplied interface and port
TIBCO-BW-ADMIN-500454	TEA interface and port of the BWAgent [{0}] conflicted with

Message Codes	Message
	the supplied interface and port
TIBCO-BW-ADMIN-500455	Interface and the port of listen URL of the BWAgent [{0}] conflict with the supplied interface and port
TIBCO-BW-ADMIN-500456	Interface and the port of listen URL of the BWAgent [{0}] conflict with the supplied interface and port
TIBCO-BW-ADMIN-500460	Archive was created with a newer version of the product [{0}] and cannot be installed into the current version [{1}]. Upgrade the product and try again
TIBCO-BW-ADMIN-500461	Invalid archive for this product version. Only archives with single modules are supported.
TIBCO-BW-ADMIN-500462	Archive was created with the different BusinessWorks Edition and cannot be installed in the current product. Upgrade the product and try again.
TIBCO-BW-ADMIN-500463	Host and port combination [{0}:{1}] is already in use.
TIBCO-BW-ADMIN-500464	Invalid interface name. No IP address was found for host [{0}]
TIBCO-BW-ADMIN-500465	Failed to enable statistics collection for Application [{0}:{1}], {2}
TIBCO-BW-ADMIN-500466	Failed to disable statistics collection for Application [{0}:{1}], {2}
TIBCO-BW-ADMIN-500467	Invalid profile file from [{0}].
TIBCO-BW-ADMIN-500469	Invalid property [{0}] is unset.
TIBCO-BW-ADMIN-500470	Unable to connect to Mashery[{0}]
TIBCO-BW-ADMIN-500471	Mashery configuration file does not exist.

Message Codes	Message
TIBCO-BW-ADMIN-500504	Failed to register TEA Agent [{0}] with TEA server [{1}], {2}
TIBCO-BW-ADMIN-000001	BW6_INFR_UNEXP_EXCP_FORMAT = {0}
TIBCO-BW-ADMIN-000100	Failed to create the application [{0}] in the data store
TIBCO-BW-ADMIN-000101	Failed to read archive file for [{0}] from data store
TIBCO-BW-ADMIN-000200	Internal error: {0}

BWAgent Logging

The BWAgent creates a log file called `bwagent.log` that is saved in the *BW_HOME*`/logs` folder.
The default log is configured as a daily roller appender and is automatically compressed as a ZIP file. The default logging level is INFO. The Logback configuration file is *BW_HOME*`/bin/bwagent-logback.xml`. When the BWAgent is started, the `bwagent.log` log file is created. If the BWAgent could not be started, informational messages are displayed to help track down the cause. For the default logging level, the BWAgent displays messages similar to the following.

```
C:\work\BW-v25\bw\6.6\bin>bwagent
TIBCO ActiveMatrix BusinessWorks version 6.6.0, build V25, 2019-07-09
13:45:39.405 WARN  [main] org.eclipse.jetty.server.Server -
ErrorPageMapper not supported for Server level Error Handling
TIBCO-BW-AGENT-300002: BusinessWorks Agent started successfully.
```

To view and change the logging level for BWAgent, follow these steps. The BWAgent does not have to be restarted for the logging level change to take effect.

Procedure

1. Open *BW_HOME*`/bin/bwagent-logback.xml` file in a text editor. Change the ROOT level setting at the end of the file as needed.

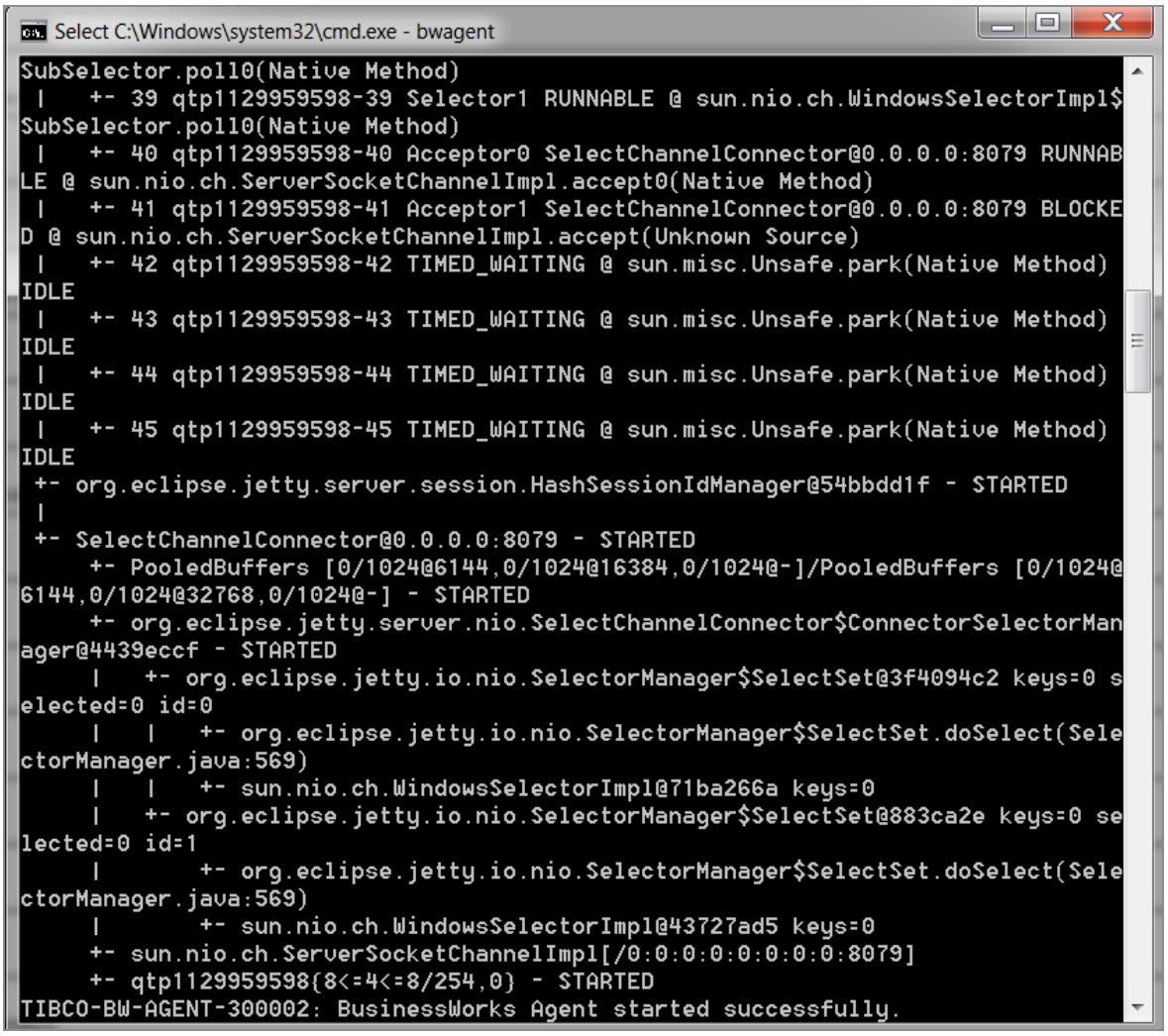
```
<root level="INFO">
    <appender-ref ref="STDOUT" />
    <appender-ref ref="FILE" />
```

2. Optional. By default, the Jersey and Jetty based loggers have the WARN log level. To modify the log level, update the following loggers:

```
<logger name="org.eclipse.jetty" level="DEBUG" additivity="false">
    <appender-ref ref="STDOUT" />
    <appender-ref ref="FILE" />
  </logger>

  <logger name="org.glassfish.jersey" level="DEBUG"
additivity="false">
    <appender-ref ref="STDOUT" />
    <appender-ref ref="FILE" />
  </logger>
```

3. To get the BWAgent log file at the BWAdmin console, open a terminal and start BWAdmin if it is not already started. Type: `getlogfile agent`

   The log file contains messages for each activity. The following illustration displays messages BWAgent startup messages.



You can also view the log file in a text editor.

HTTP Logging

When Jetty servers used in shared resources receive requests, logs are created to capture all attempts to access the servers.

By default, logging is disabled. You can enable HTTP logging from TIBCO Business Studio for BusinessWorks and from a deployed application.

To enable and test logging from TIBCO Business Studio for BusinessWorks in the **HTTP Connector Resource** shared resource:

1. On the **Advanced** tab, select the **Enable Access Logs** checkbox. Then select **Logging**

Configuration.

2. Navigate to `bw\<Release_Number>\config\design\logback` and select `logback_leveldebug.xml`.
3. Check the console logs to make sure that they display content.

Note: If you use REST service binding to create the HTTP Connector resource at runtime, set the `bw.engine.http.jetty.accesslogs.enable` system property to `true` to enable Jetty logs.

To enable and test logging from a deployed application:

1. Deploy the application on a TEA server.
2. Enable the debug logging for `appnode`.
3. Check the `appnode.log` file.
4. Check for log contents similar to the following:

```
++++++++++++++++++++
2017-01-06 16:26:17.609 INFO [bwResourceHTTPConnector.qtp-112]
com.tibco.bw.http.jetty.accesslogger - [HTTP Connector: mpandav-
t450:6565] mpandav-t450
192.168.56.1 - - [06/Jan/2017:16:26:17 +0530] "GET /?hello HTTP/1.1" 200
41 "-"
"Jakarta Commons-HttpClient/3.1" - 32 32
++++++++++++++++++++
```

Note: To populate the HTTP logs, enable debug logging for your application.

For more information on enabling HTTP access logging, see "HTTP Connector" in the *TIBCO ActiveMatrix BusinessWorks™ Bindings and Palette Reference* guide.

Note: The **Enable Access Logs** checkbox is supported only from ActiveMatrix BusinessWorks 6.3.2 version onwards. To enable HTTP access logging for applications created on previous versions of ActiveMatrix BusinessWorks, set the `bw.plugin.http.jetty.accesslogs` property to `true`.

Viewing Log Files from the Admin UI

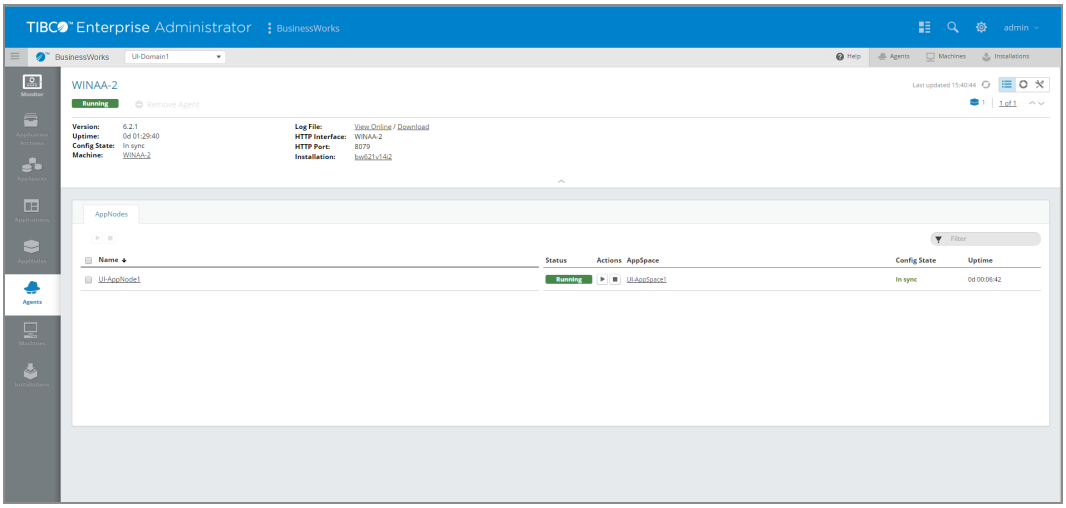
AppNode and BWAgent log files are available from the Admin UI.

Before you begin

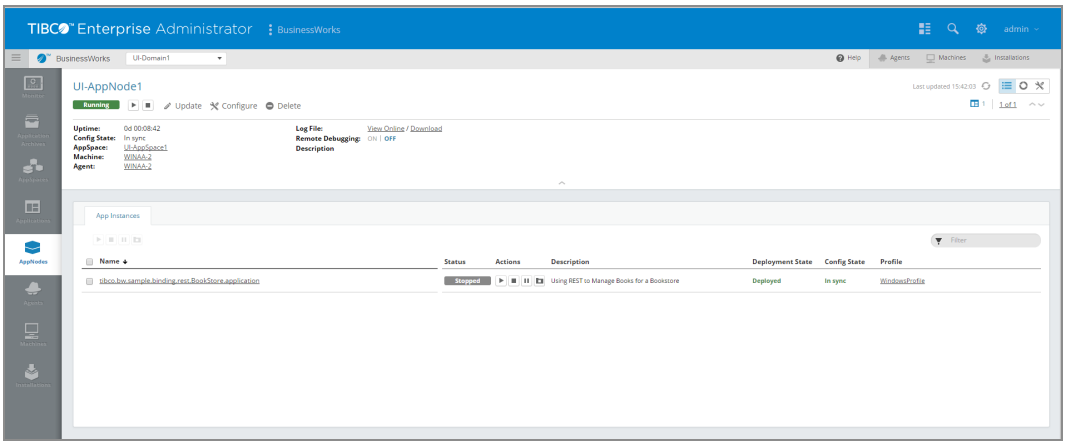
- The Admin UI is running.
- At least one AppNode must be started.

Procedure

1. Open the Admin UI.
2. Select the BusinessWorks product icon on the home page and click a domain.
3. To view the log file for the BWAgent, click the Agents icon on the left, then drill into the BWAgent by clicking the BWAgent name on the Agents page. The page for the selected BWAgent is displayed:



4. Click the Log File **View Online** or **Download** link. View Online displays the BWAgent log file in a new browser window. Download downloads the file to your machine. This log file is created by the BWAgent using the logging configuration provided in the `logback.xml` file. For more information, see BWAgent Logging.
5. To view the log file for an AppNode, click the AppNodes icon on the left, then choose an AppNode by clicking the AppNode name on the AppNode page. The page for the selected AppNode is displayed:



6. Click the Log File **View Online** or **Download** link. View Online displays the AppNode log file in a new browser window. Download downloads the file to your machine. For more information about AppNode logging, see AppNode Logging.

Fault Tolerance

Fault tolerance is the ability of the system to continue processing requests when an unexpected failure occurs on one of the AppNodes in the AppSpace.

Fault tolerance is supported only at the AppNode level. When an unexpected failure occurs on one of the AppNodes in an AppSpace, the application is no longer be available on that AppNode. However, the fault tolerance configuration enables the application to continue to provide service and process requests through the other AppNodes in the AppSpace. Depending on the activation mode selected for the application (See Activation Modes), the fault tolerance configuration can behave in the following ways:

- Distributes the incoming request load among other AppNodes in the AppSpace.
- If an AppNode that has an application in active state fails, another AppNode that has an application in the passive (stand-by) state takes over and starts processing requests.
- The check-pointed job data from an application in the failed AppNode can be recovered by another AppNode.
- If an application is in the standby or disabled mode, the status in the **Components** tab in Admin UI changes to Standby, and the starter state displayed in the command line changes to Not Active. For more information on retrieving the list of components, see Retrieving list of components in an Application.

ActiveMatrix BusinessWorks fault tolerance feature can be classified into the following types: Managed Fault Tolerance and Non-managed Fault Tolerance.

Managed Fault Tolerance

In managed fault tolerance, when an AppNode fails, the application on another AppNode takes over automatically. The AppNodes in an AppSpace are aware of each other's existence and the engines collaborate to provide fault tolerance.

The managed fault tolerance requires:

- The engine persistence mode (`bw.engine.persistenceMode`) to be set to type group. The persistence mode of type group requires both database and group provider

configurations. For more information, see Engine Persistence Modes.

- A minimum of two AppNodes in an AppSpace.

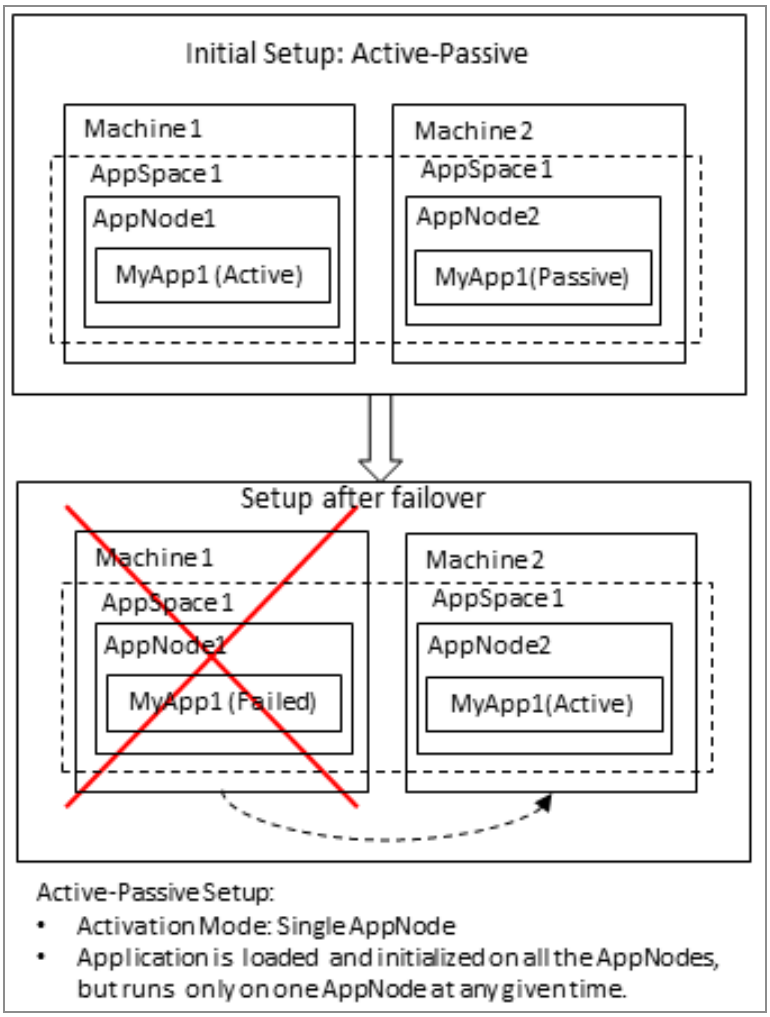
The managed fault tolerance configuration supports both the application activation modes - **Single AppNode** and **Multiple AppNodes**. For more information, see Application Activation Modes.

The following table lists the managed fault tolerance features available for each of the activation modes.

Managed Fault Tolerance Features for Application Activation Modes

Single AppNode (Active-Passive)	Multiple AppNode (Active-Active)
The incoming requests are only processed by an AppNode where the application is in an active state.	The incoming requests can be processed by any AppNode since the application is active in all AppNodes.
On failure of an AppNode that has the application in an active state, it automatically enables the application in another AppNode to take over and start processing requests.	On failure of an AppNode, other AppNodes continue to process new requests.
The check pointed data from an application in the failed AppNode can be recovered by the application that is automatically enabled in another AppNode.	The check pointed data from an application in the failed AppNode can be automatically recovered by another AppNode.

Fault-tolerant Fail-over



Non-managed Fault Tolerance

In non-managed fault tolerance, the AppNodes in an AppSpace are not aware of each other's existence and there is no collaboration between the engines. Consequently, if an AppNode fails, then another AppNode in the AppSpace does not take over.

The non-managed fault tolerance requires:

- The engine persistence mode (`bw.engine.persistenceMode`) to be set to type datastore. The persistence mode of type datastore requires database configurations. For more information, see Engine Persistence Modes.
- If there are multiple AppNodes in the AppSpace, then each AppNode must be configured with a unique database configuration. An AppNode specific database

configuration is stated through the AppNode `config.ini` file.

The application activation mode is not applicable in non-managed fault tolerance configuration. That is, the application activation modes, Single AppNode or Multiple AppNodes, are not supported in the non-managed fault tolerance. For more information, see Activation Modes. The application is activated in all AppNodes. However, unlike the managed fault tolerance, the other AppNodes in the AppSpace are not aware of each other.

The following features are available for non-managed fault tolerance:

- The incoming requests can be processed by any AppNode since the application is active in all AppNodes.
- On failure on an AppNode, other AppNodes continue to process new requests.
- An application can have a checkpoint. However, on the failure of an AppNode, another AppNode does not recover the check-pointed data.

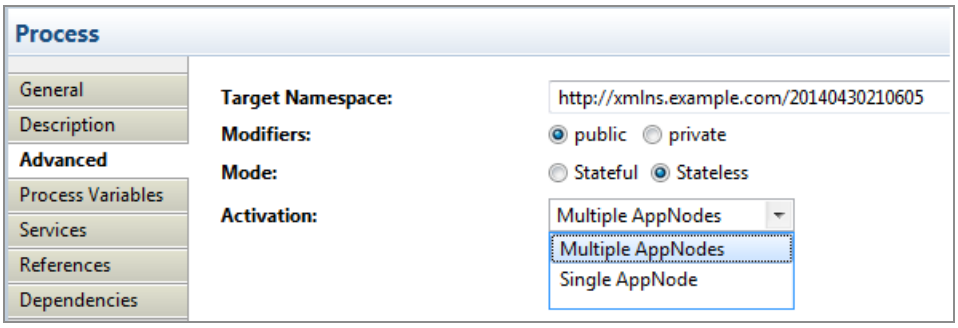
Application Activation Modes

Activation mode for an application defines the way an application is loaded, initiated, and started when deployed to an AppSpace.

By choosing the appropriate activation mode, the component in an application module can be configured to be active in multiple AppNodes or in a single AppNode. When configured to be active in a single AppNode, the component is active in a single AppNode and in a passive state in all the other AppNodes of the AppSpace.

A component in an active state is loaded, initialized, and enabled to process new events. A component in a passive state is loaded, initialized. However it is not enabled to process new events instead, it is in a stand-by mode. When a component is active in multiple AppNodes, it is considered to be Active-Active. And when a component is active in a single AppNode and passive in other AppNodes, it is considered to be Active-Passive.

The component's activation mode is determined by the process that implements the component. At design-time, the process that is implemented by a component in an application module can be configured to be active in a single AppNode (Active-Passive) or in multiple AppNodes (Active-Active). To do so, set the **Activation** field on the **Advanced** tab of the Properties view as shown:



Note: If the activation mode for a process is set to Multiple AppNodes, then the activation mode for all the subprocesses and child processes (other processes that are called by the process or one of its child processes) must be set to Multiple AppNodes as well. Failure to do so can result in a design-time validation error.

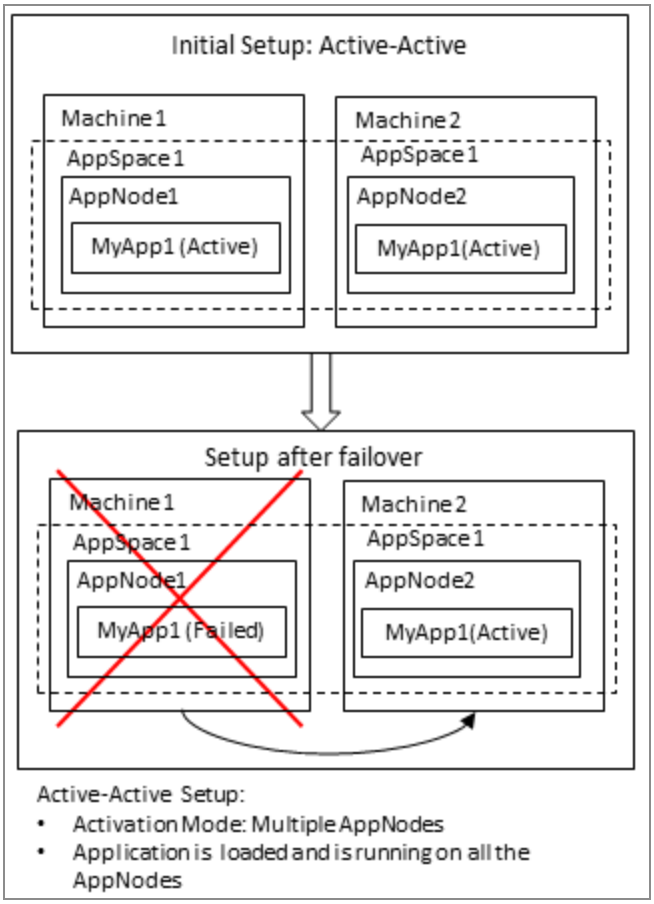
Important: Enabling the activation mode for an application requires both design-time and runtime configurations as described in the following sections.

Active-Active (Multiple AppNodes)

In active-active mode, an application is loaded, initiated, and ready to run on all the AppNodes in the AppSpace. Enabling an application for active-active mode requires both design-time and runtime configurations. Perform the following steps to configure an application for active-active mode:

- **Design-time**: Set the activation mode of the component process to Multiple AppNodes. The `Activation` field is configurable from the **Advanced** tab of the Properties view for the process.
- **Runtime**: The engine persistence mode must be set to group to enable the engines running in the AppNodes to be aware of the existence of other engines in the AppSpace and provide a managed fault tolerance feature. See Engine Persistence Modes for details about persistence modes. For more information about how to configure the runtime, see Fault Tolerance.

Active-Active Mode



Active-Passive (Single AppNode)

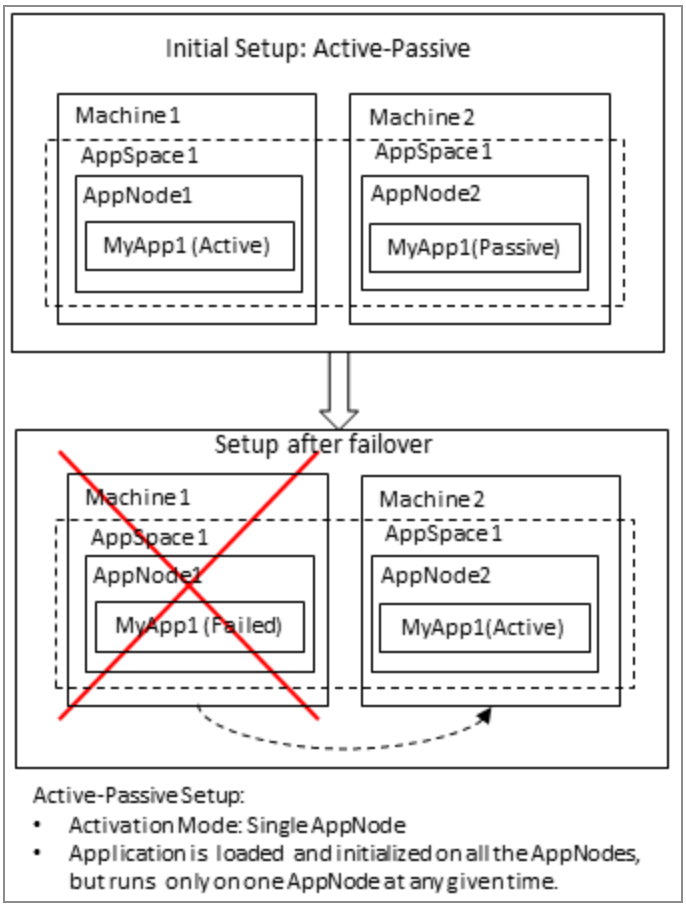
In active-passive mode, an application is loaded on all the AppNodes in the AppSpace, and is activated to run on a single AppNode at any given time. Enabling an application for active-passive mode requires both design-time and runtime configurations. Perform the following steps to configure an application for active-passive mode:

- **Design-time**: Set the activation mode of the component process to Single AppNode. The `Activation` field is configurable from the **Advanced** tab of the Properties view for the process.
- **Runtime**: The engine persistence mode must be set to group to enable the engines running in the AppNodes to be aware of the existence of other engines in the AppSpace and provide a managed fault tolerance feature.

  For details about persistence modes, see Engine Persistence Modes.

For details about how to configure the runtime, see Fault Tolerance.

Active-Passive Mode



Note: When an application is running in active-passive mode on multiple AppNodes, stopping an application instance that is active in an AppNode does not trigger the passive application instances in other AppNodes to become active. The only way a passive application instance in another AppNode becomes active is when the AppNode that contains the active application is ended.

Engine Persistence Modes

Engine persistence mode defines whether engines located on one or more physical machines collaborate with each other or work independently. There are four modes of engine persistence: memory, datastore, group, and ftgroup.

Engine persistence is defined at the AppSpace level and is set to memory by default. To change the engine persistence mode, run the utility to set the persistence mode property `bw.engine.persistenceMode` to datastore,group or ftgroup.

Engine Persistence Mode: Memory

`bw.engine.persistenceMode`=memory: By default, the engine persistence mode is set to memory. In the memory mode, there is no persistence and the engines are unaware of the existence of each other. As a result, there is no collaboration between engines.

Engine Persistence Mode: Datastore

`bw.engine.persistenceMode`=datastore: The datastore mode uses a database to provide persistence and requires a database configuration. The database connection configuration can be specified at the AppNode level. For more information, see Configuring Database for the Engine. The datastore engine persistence mode is required for persistence features such as checkpointing and module shared variables.

Engine Persistence Mode: Group

`bw.engine.persistenceMode`=group: The group mode uses a database and a group provider to provide persistence and collaboration between the AppNodes. In the group mode, the engines are aware of each other's existence and they can collaborate and work together to enable features such as checkpointing and managed fault tolerance. The configuration details must be specified at the AppSpace level. For more information, see Configuring Database for the Engine and Configuring the Engine for Group Persistence Mode.

Note: In group mode, the engine requires both DB and TIBCO Enterprise Message Service™ (EMS) or TIBCO FTL® as mandatory infrastructure requirement. Along with the data persistence and collaboration, AppNode uses the database for internal functions too. So ensure that the database is always available.

Engine Persistence Mode: FTGroup

`bw.engine.persistenceMode`=ftgroup:
The ftgroup mode does not use a database, and disregards the application activation mode, but does use a group provider to provide minimal collaboration between the AppNodes. Only one AppNode run the applications, while the other AppNodes are standing

by, to take over in the event of a failure of the active AppNode. When configured for the ftgroup persistence mode, the engine requires a group provider such as TIBCO Enterprise Message Service™ (EMS) or TIBCO FTL®, to be configured. Also, note that, since the ftgroup mode does not use a database, this mode does not support checkpointing. For more information, see Configuring the Engine for FTGroup Persistence Mode.

Configuring a Database for the Engine

Checkpoint activity and other persistence features require the Engine Persistence Mode (`bw.engine.persistenceMode`) to be configured for a `datastore` or `group` mode. When the Engine Persistence Mode property is configured for `datastore` or `group` mode, the engine requires a database configuration.

Procedure

1. Scripts for creating the engine database for various database types are at *BW_HOME*`/config/dbscripts/engine`. Based on whether the Engine Persistence Mode property is configured for `datastore` mode or `group` mode, complete one of the following steps:
    a. If the Engine Persistence Mode property is set to `datastore` mode, run the bundled scripts `create.sql` and `create-scp.sql` to create the engine database.
    b. If the Engine Persistence Mode property is set to `group` mode, run the bundled scripts `create.sql` and `create-dcp.sql` to create the engine database.

    > **Note:** The `create.sql`, `create-scp.sql`, and `create-dcp.sql` scripts are available for each vendor directory in the *{BW_HOME}*`\config\dbscripts\engine` directory.

2. To change the Engine Persistence Mode, run the utility to set the Persistence Mode property, `bw.engine.persistenceMode` to datastore or group, and then configure the engine database connection details.

    ```
    bw.engine.persistenceMode=[datastore | group]
    ```

> **Note:** Before updating the AppSpace configuration, you must stop the AppSpace if it is running.

The database connection configuration can be specified at the AppSpace or the AppNode level. The database connection details specified at the AppSpace level are applied to all AppNodes within the AppSpace. The configuration specified at the AppNode level takes precedence over the configuration specified at the AppSpace level.

When the Engine Persistence Mode property is set to group, the database connection configuration must be specified only at the AppSpace level.

When the Engine Persistence Mode property is set to datastore, the database connection configuration cannot be shared by two or more AppNodes in the same AppSpace. As a result, the database connection configuration can be specified at the AppSpace level only if the AppSpace contains a single AppNode. For an AppSpace that contains two or more AppNodes, each AppNode requires a unique database and the database connection configuration must be specified at the AppNode level.

3. To set database configuration properties at the AppSpace level, follow these steps:

> **Note:** Ensure you are using a different database instance for each AppSpace. To do this with a single database, create a tablespace or schema for each AppSpace.

   a. Copy the existing AppSpace `config.ini` file (in the root of the AppSpace folder), or the AppSpace `config.ini` template file `appspace_config.ini_template` (in *`BW_HOME`*`/config/`) to a temporary location.

   b. Edit the Engine Persistence Mode property, `bw.engine.persistenceMode`, and set it to datastore or group.

```
bw.engine.persistenceMode=[datastore | group]
```

   c. Configure the following database connection properties in the **BWEngine Database Configuration** section of the `config.ini` file:

```
#------------------------------------------------------------
```

```
-----------------
# Section:  BW Engine Database Configuration.
#
# The properties in this section are applicable to the BW
Engine database.
# All properties in this section are required when the BW
Engine
# property "bw.engine.persistenceMode" set to "datastore" or
"group".
# ------------------------------------------------------------
------------------
# BW Engine Database Driver.
bw.engine.db.jdbcDriver=org.postgresql.Driver

# BW Engine Database URL.

bw.engine.db.url=jdbc:postgresql://<servername>:<portnumber>/<
dbname>

# BW Engine Database User Name.
bw.engine.db.userName=user1

# BW Engine Database User Password.
bw.engine.db.password=

# BW Engine Database Connection Pool Size.
bw.engine.db.maxConnections=15
```

When setting the password property (`bw.engine.db.password`), the default format is plain text. Run the command `bwadmin obfuscate`, or the command `bwobfuscator`, from the command line to encrypt the password; use the generated encrypted text as the password.

Note: The BWAdmin `bwenginedb` command displays BWEngine datastore configuration settings.

4. To set the database for datastore mode at the AppNode level, follow these steps:
    a. Copy the existing AppNode `config.ini` file (in the root of the AppNode folder) to a temporary location.
    b. Set Engine Persistence Mode property, `bw.engine.persistenceMode` to datastore and configure the engine database connection details.

```
bw.engine.persistenceMode=[datastore]
```

c. Configure the engine database connection properties in the **BWEngine datastore configuration** section of the `config.ini` file. By default, the AppNode `config.ini` file does not contain these properties. Copy these properties from the AppSpace `config.ini` template file, `appspace_config.ini_template`, in *`BW_HOME`*`/config` to the AppNode `config.ini` file and provide the database connection details.

5. Use one of the following `config` admin commands to push the configuration to the AppSpace or the AppNode:
   - AppSpace:

     ```
     bwadmin[admin] > config -d myDomain -a myAppSpace -cf
     <temporaryLocation>/config.ini
     ```

   - AppNode:

     ```
     bwadmin[admin]> config -d myDomain -a myAppSpace -n myAppnode -
     cf <temporaryLocation>/config.ini
     ```

6. Restart the AppSpace.

> **Note:** Before you clean the engine database, ensure that you have backed up all important data.

7. To clean the engine database that is configured for **datastore mode**, run the `drop.sql` and `drop-scp.sql` scripts. If the engine database is configured for **group mode**, run the `drop.sql` and `drop-dcp.sql` scripts.

Result

You used the BWAdmin command line to set the database configuration property. You can also use the Admin UI to set this property. See the following topics from the *TIBCO ActiveMatrix BusinessWorks™ Administration* guide.

- Editing an AppSpace Configuration
- Editing an AppNode Configuration

Configuring the Engine for Group Persistence Mode

The managed fault tolerance feature requires the engine persistence mode to be configured for the group mode. The group mode also supports the Checkpoint activity and other persistence features. When configured for the group persistence mode, the engine requires both a database and a group provider, such as TIBCO Enterprise Message Service™ (EMS) or TIBCO FTL®, to be configured.

Refer to the following topics for instructions about setting TIBCO EMS or TIBCO FTL as the group provider technology for the engine:

- Configuring EMS as the Group Provider for Engine
- Configuring FTL as the Group Provider for Engine

Configuring TIBCO FTL® as the Group Provider for Engine

Follow these steps to configure the engine for group persistence mode, and to set TIBCO FTL as the group provider technology.

Note: Use of TIBCO FTL with ActiveMatrix BusinessWorks for configuring BWAgent and for configuring group provider for engine does not require TIBCO FTL licenses.

Before you begin

- See the ActiveMatrix BusinessWorks readme for the version of TIBCO FTL that is supported with the version of ActiveMatrix BusinessWorks 6.x you are using.
- Ensure you have installed FTL client libraries. For more information, see Integrating with TIBCO FTL in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.
- These steps are only applicable if you are not using TIBCO FTL as the BWAgent transport.
- If you are installing TIBCO FTL after you have already installed ActiveMatrix BusinessWorks, set the `tibco.env.FTL_HOME` variable in the `bwcommon.tra` file. You can find this file in the bin folder at *`BW_HOME`*`\bin` for Windows, or *`${BW_HOME}`*`/bin` for

Unix.

1. Install TIBCO FTL. For instructions, see the *TIBCO FTL® Installation* guide.
2. Start the FTL Realm server by running the `./tibrealmserver -ht <hostIP>:<port>` FTL command.

   ```
   ./tibrealmserver -ht <hostIP>:<port>
   ```

3. Run the following FTL command to populate data in the `bwadmin_ftlrealmserver.json` template file, in the `config` folder at *BW_HOME*`/config`:

   ```
   ./tibrealmadmin -rs <realmserverurl> -ur <PATH of bwadmin_ftl_realmserver.json>
   ```

Note: For instructions about configuring an FTL backup server for high availability, see "Configuring Backup Realm Servers for Fault Tolerance" in the *TIBCO FTL® Administration* guide.

Procedure

1. Create the engine database by running the bundled scripts `create.sql`,`create-scp.sql` and `create-dcp.sql`. Scripts for creating the engine database for various database types are located at *BW_HOME*`/config/dbscripts/engine`. The engine directory contains folders for the supported database types, and scripts for each database can be found in the respective folders.
2. Set the Engine Persistence Mode property (`bw.engine.persistenceMode`) to group and configure the engine group configuration.
   a. Copy the existing AppSpace `config.ini` template file `appspace_config.ini_template` (in *BW_HOME*`/config`) to the root of the AppSpace folder, or a temporary location, and rename the file as `config.ini`.
   b. Edit the ActiveMatrix BusinessWorks Engine Persistence Mode property, `bw.engine.persistenceMode`, and set it to group.

      ```
      bw.engine.persistenceMode=group
      ```

   c. Specify the group name and group provider technology as ftl in the `config.ini` file. The group name is optional and it defaults to domain and AppSpace names separated by an underscore (_).

Note: Ensure you are using a different database instance for each AppSpace.

```
# ------------------------------------------------------------
-----------------
# Section:  BW Engine Group Configuration.
#
# The properties in this section are applicable to the BW
Engine group.
# Some of the properties in this section are required when the
BW Engine
# property "bw.engine.persistenceMode" is set to "group".
# ------------------------------------------------------------
-----------------
# BW Engine Group Name.  This is an optional property and it
specifies name of
# the BW engine group.  If this property is not specified,
then the group name
# defaults to "Group_<DomainName>_<AppSpaceName>".
#bw.engine.groupName=mytestgroup

# BW Engine Group Connection Provider Technology.  This is a
required property
# when the persistenceMode is set to "group"
(bw.engine.persistenceMode=group)
# and it specifies the BW Engine group communication
technology.  The only
# supported values are "ems" and "ftl".  The group connection
provider technology property
# requires additional configuration.  See section "Configuring
the Engine for Group Persistence Mode"
# for additional configuration.
bw.engine.groupProvider.technology=ftl
```

d. Specify the group provider configuration:

```
# ------------------------------------------------------------
-----------------
# Section:  BW Engine Group Connection Provider FTL
Configuration.
```

```
#
# Some of the properties in this section are required when the
BW Engine Group
# Connection Provider Technology property
"bw.engine.groupProvider.technology"
# value is set to "ftl"
# ------------------------------------------------------------
------------------
# BW Engine Group Connection Provider FTL Realm Server.  This
property is required if
# the group provider technology is "ftl".
bw.engine.groupProvider.ftl.realmserver=http://localhost:8080

# BW Engine Group Connection Provider FTL Realm client
username
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.username=

# BW Engine Group Connection Provider FTL Realm client
password
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.password=

# BW Engine Group Connection Provider FTL application
identifier
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.appinstance.id=bwadmin-endpoint

# BW Engine Group Connection Provider FTL secondary realm
server
# This property is optional.
#bw.engine.groupProvider.ftl.secondaryserver=

# BW Engine Group Connection Provider FTL group name
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.groupname=

# BW Engine Group Connection Provider FTL application name
```

```
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.appname=bwadmin

# BW Engine Group Connection Provider FTL publish endpoint
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.publish.endpoint=bwadmin-endpoint

# BW Engine Group Connection Provider FTL application name
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.subscribe.endpoint=bwadmin-
endpoint
```

When setting the password property (`bw.engine.groupProvider.ftl.password`), the default format is plain text. Run the command `bwadmin obfuscate`, or the command `bwobfuscator`, from the command line to encrypt the password; use the generated encrypted text as the password.

3. **Optional.** If you have saved the `config.ini` file to a temporary location, ensure you copy it to the AppSpace root folder in *`BW_HOME`*`/domains/defaultdomain/appspaces/defaultappspace`.
4. Use the following `config` admin command to push the configuration to the AppSpace:

```
bwadmin[admin] > config -d myDomain -a myAppSpace -cf
<temporaryLocation>/config.ini
```

Configuring EMS as the Group Provider for Engine

Follow these steps to configure the engine for group persistence mode, and to set TIBCO EMS as the group provider technology.

Procedure

1. Create the engine database by running the bundled scripts `create.sql`,`create-scp.sql` and `create-dcp.sql`. Scripts for creating the engine database for various

database types are in `BW/Home/config/dbscripts/engine`. The engine directory contains folders for the supported database types, and scripts for each database can be found in the respective folders.

2. Set the Engine Persistence Mode property (`bw.engine.persistenceMode`) to group and configure the engine group configuration.
    a. Copy the existing AppSpace `config.ini` template file `appspace_config.ini_template` (in *`BW_HOME`*`/config`) to the root of the AppSpace folder, or a temporary location, and rename the file as `config.ini`.
    b. Edit the ActiveMatrix BusinessWorks Engine Persistence Mode property, `bw.engine.persistenceMode`, and set it to group.

       Follow these steps to configure the engine for group persistence mode, and to set TIBCO Enterprise Message Service™ (EMS) as the group provider technology.

       ```
       bw.engine.persistenceMode=group
       ```

    c. Specify the group name and group provider technology in the `config.ini` file. The group name is optional and it defaults to domain and AppSpace names separated by an underscore (_).

       Note: You can use a different database instance for each AppSpace. Alternatively, you can use a single database instance for multiple AppSpaces if you create a tablespace or schema for each one.

       ```
       # ------------------------------------------------------------
       -----------------
       # Section:  BW Engine Group Configuration.
       #
       # The properties in this section are applicable to the BW
       Engine group.
       # Some of the properties in this section are required when the
       BW Engine
       # property "bw.engine.persistenceMode" is set to "group".
       # ------------------------------------------------------------
       -----------------
       # BW Engine Group Name.  This is an optional property and it
       specifies name of
       ```

```
# the BW engine group.  If this property is not specified,
then the group name
# defaults to "Group_<DomainName>_<AppSpaceName>".
#bw.engine.groupName=mytestgroup

# BW Engine Group Connection Provider Technology.  This is a
required property
# when the persistenceMode is set to "group"
(bw.engine.persistenceMode=group)
# and it specifies the BW Engine group communication
technology.  The only
# supported values are "ems" and "ftl".  The group connection
provider technology property
# requires additional configuration.  See section "Configuring
the Engine for Group Persistence Mode"
# for additional configuration.
bw.engine.groupProvider.technology=ems
```

d. Specify the group provider configuration:

```
# ------------------------------------------------------------
------------------
# Section:  BW Engine Group Connection Provider EMS
Configuration.
#
# Some of the properties in this section are required when the
BW Engine Group
# Connection Provider Technology property
"bw.engine.groupProvider.technology"
# value is set to "ems".
# ------------------------------------------------------------
------------------
# BW Engine Group Connection Provider EMS URL.  This property
is required if
# the group provider technology is "ems".
bw.engine.groupProvider.qin.EMSServerUrl=tcp://localhost:7222

# BW Engine Group Connection Provider EMS User Name.  This
property is required
# if the group provider technology is "ems".
bw.engine.groupProvider.qin.EMSUserName=admin
```

```
# BW Engine Group Connection Provider EMS User Password.  This
property is
# required if the group provider technology is "ems".
bw.engine.groupProvider.qin.EMSPassword=

# BW Engine Group Connection Provider EMS Member Prefix.  This
property is
# optional and the default value is "EMSGMS".
#bw.engine.groupProvider.qin.EMSPrefix=EMSGMS

# BW Engine Group Connection Provider EMS Recovery Timeout in
ms.  This
# property is optional and the default value is "5000" ms.
#bw.engine.groupProvider.qin.EMSRecoveryTimeout=5000

# BW Engine Group Connection Provider EMS Recovery Attempt
Delay in ms.  This
# property is optional and the default value is "500" ms.
#bw.engine.groupProvider.qin.EMSRecoveryAttemptDelay=500

# BW Engine Group Connection Provider EMS Recovery
AttemptCount.  This
# property is optional.
#bw.engine.groupProvider.qin.EMSRecoveryAttemptCount=

# BW Engine Group Connection Provider EMS Connect Attempt
Count.  This property
# is optional.
#bw.engine.groupProvider.qin.EMSConnectAttemptCount=

# BW Engine Group Connection Provider EMS Connect Attempt
Delay in ms.  This
# property is optional.
#bw.engine.groupProvider.qin.EMSConnectAttemptDelay=
```

When setting the password property (`bw.engine.groupProvider.qin.EMSPassword`), the default format is plain text. Run the command `bwadmin obfuscate`, or the command `bwobfuscator`, from the command line to encrypt the password; use the generated encrypted text as the password.

3. **Optional**. The following properties are available for EMS SSL configuration.

```
EMS SSL Configuration
```

```
#client identity consisting of the certificate,
#private key and optionally extra issuer certificates can be
included into a single data block using PKCS12.
#Keystore or Entrust Store encodings
#bw.engine.groupProvider.ems.ssl.trust.identity=

#The set of Trusted Certificates represents all trusted issuers of
the server certificate.
#It must be specified by the client application unless the host
certificate verification is completely disabled.
#bw.engine.groupProvider.ems.ssl.trust.certlocation=

#EMS SSL connection trust password. This
#property is required if the JMS server protocol is ssl. The
password may
#be clear text or supplied as an obfuscated string.
#bw.engine.groupProvider.ems.ssl.trust.password=

#trusted certificate commonname must match the ems server hostname
if set to false
#bw.engine.groupProvider.ems.ssl.disable.verifyHostName=

#client and server certificates must match if set to false
#bw.engine.groupProvider.ems.ssl.trust.disable.verifyHost=
```

4. **Optional.** If you have saved the `config.ini` file to a temporary location, ensure you copy it to the AppSpace root folder in *`BW_HOME`*`/domains/defaultdomain/appspaces/defaultappspace`.
5. Use the `config` admin command to push the configuration to the AppSpace: `bwadmin [admin] > config -d myDomain -a myAppSpace -cf <`*`temporaryLocation`*`>/config.ini`.

Configuring the Engine for FTGroup Persistence Mode

In the managed fault tolerance ftgroup mode, only one AppNode runs the application, and the other AppNodes are standing by.

When the `bw.engine.ftgroup.lbmode` property is set to true in the `config.ini` file at the AppSpace level, all processes having **Activation mode** set as Multiple AppNodes in TIBCO Business Studio™, run on all `bwengines` in the group.

Note: By default, `bw.engine.ftgroup.lbmode` is set to false.

When you want to elect an AppNode as a leader AppNode, then set the `bw.engine.use.weighted.node` property to true at an AppSpace level. For more information about the `bw.engine.use.weighted.node` property, see BW Engine ftgroup Properties.

Note: By default, the `bw.engine.use.weighted.node` property is set to false.

The ftgroup mode does not support checkpointing. When configured for the ftgroup persistence mode, the `bwengine` requires the group provider, such as TIBCO Enterprise Message Service™ (EMS) or TIBCO FTL®, to be configured.

For instructions about setting TIBCO EMS or TIBCO FTL as the group provider technology for the engine, see the following topics:

- Configuring EMS as the FTGroup Provider for Engine
- Configuring TIBCO FTL® as the FTGroup Provider for Engine

Configuring TIBCO FTL® as the FTGroup Provider for Engine

Follow these steps to configure the engine for ftgroup persistence mode, and to set TIBCO FTL as the group provider technology.

Note: Use of TIBCO FTL with TIBCO ActiveMatrix BusinessWorks™ for configuring BWAgent and for configuring group provider for engine does not require TIBCO FTL licenses.

Before you begin

- See the ActiveMatrix BusinessWorks™ readme for the version of TIBCO FTL that is supported with the version of ActiveMatrix BusinessWorks 6.x you are using.
- Ensure you have installed FTL client libraries. For more information, see "Integrating with TIBCO FTL" in the *TIBCO ActiveMatrix BusinessWorks™ Installation* guide.

- These steps are only applicable if you are not using TIBCO FTL as the BWAgent transport.
- If you are installing TIBCO FTL after you have already installed ActiveMatrix BusinessWorks, set the *tibco.env.FTL_HOME* variable in the `bwcommon.tra` file. You can find this file in the bin folder at *BW_HOME*\bin for Windows, or *${BW_HOME}*/bin for Unix.

1. Install TIBCO FTL. For instructions, see the *TIBCO FTL® Installation* guide.
2. Start the FTL Realm server by running the ./tibrealmserver -ht <*hostIP*>:<*port*> FTL command.

   ```
   ./tibrealmserver -ht <hostIP>:<port>
   ```

3. Run the following FTL command to populate data in the bwadmin_ ftlrealmserver.json template file, in the config folder at *BW_HOME*`/config`:

   ```
   ./tibrealmadmin -rs <realmserverurl> -ur <PATH of bwadmin_ftl_realmserver.json>
   ```

Note: For instructions about how to configure an FTL backup server for high availability, see "Configuring Backup Realm Servers for Fault Tolerance" in the *TIBCO FTL® Administration* guide.

Procedure

1. Set the Engine Persistence Mode property (`bw.engine.persistenceMode`) to ftgroup and configure the engine group configuration.
   a. Copy the existing AppSpace `config.ini` template file `appspace_config.ini_template` (in *BW_HOME*`/config`) to the root of the AppSpace folder, or a temporary location, and rename the file as `config.ini`.
   b. Edit the ActiveMatrix BusinessWorks™ Engine Persistence Mode property, `bw.engine.persistenceMode`, and set it to ftgroup.

      ```
      bw.engine.persistenceMode=ftgroup
      ```

   c. Specify the group name and group provider technology as ftl in the `config.ini` file. The group name is mandatory.

```
# ------------------------------------------------------------
-----------------
# Section:  BW Engine Group Configuration.
#
# The properties in this section are applicable to the BW
Engine group.
# Some of the properties in this section are required when the
BW Engine
# property "bw.engine.persistenceMode" is set to "ftgroup".
# ------------------------------------------------------------
-----------------
# BW Engine Group Name.  This is a reqired property and it
specifies name of
# the BW engine group.  If this property is not specified,
then the ftgroup name
# defaults to "Group_<DomainName>_<AppSpaceName>".
#bw.engine.groupName=mytestgroup

# BW Engine Group Connection Provider Technology.  This is a
required property
# when the persistenceMode is set to "ftgroup"
(bw.engine.persistenceMode=group)
# and it specifies the BW Engine group communication
technology.  The only
# supported values are "ems" and "ftl". The group connection
provider technology property
# requires additional configuration. See section "Configuring
the Engine for Group Persistence Mode"
# for additional configuration.
bw.engine.groupProvider.technology=ftl
```

d. Specify the group provider configuration:

```
# ------------------------------------------------------------
-----------------
# Section:  BW Engine Group Connection Provider FTL
Configuration.
#
# Some of the properties in this section are required when the
BW Engine Group
# Connection Provider Technology property
"bw.engine.groupProvider.technology"
```

```
# value is set to "ftl"
# ------------------------------------------------------------
------------------
# BW Engine Group Connection Provider FTL Realm Server.  This
property is required if
# the group provider technology is "ftl".
bw.engine.groupProvider.ftl.realmserver=http://localhost:8080

# BW Engine Group Connection Provider FTL Realm client
username
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.username=

# BW Engine Group Connection Provider FTL Realm client
password
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.password=

# BW Engine Group Connection Provider FTL application
identifier
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.appinstance.id=

# BW Engine Group Connection Provider FTL secondary realm
server
# This property is optional.
#bw.engine.groupProvider.ftl.secondaryserver=

# BW Engine Group Connection Provider FTL group name
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.groupname=

# BW Engine Group Connection Provider FTL application name
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.appname=
```

```
# BW Engine Group Connection Provider FTL publish endpoint
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.publish.endpoint=

# BW Engine Group Connection Provider FTL application name
# This property is required if the group provider technology
is "ftl".
bw.engine.groupProvider.ftl.subscribe.endpoint=
```

When setting the password property (`bw.engine.groupProvider.ftl.password`), the default format is plain text. Run the command `bwadmin obfuscate`, or the command `bwobfuscator`, from the command line to encrypt the password; use the generated encrypted text as the password.

2. **Optional.** If you have saved the `config.ini` file to a temporary location, ensure you copy it to the AppSpace root folder in *`BW_HOME`*`/domains/defaultdomain/appspaces/defaultappspace`.
3. Use the `config` admin command to push the configuration to the AppSpace: `bwadmin [admin] > config -d myDomain -a myAppSpace -cf <`*`temporaryLocation`*`>/config.ini`.

Configuring EMS as the FTGroup Provider for Engine

Follow these steps to configure the engine for ftgroup persistence mode, and to set TIBCO EMS as the group provider technology.

Procedure

1. Set the Engine Persistence Mode property (`bw.engine.persistenceMode`) to ftgroup and configure the engine group configuration.
   a. Copy the existing AppSpace `config.ini` template file `appspace_config.ini_template` (in *`BW_HOME`*`/config`) to the root of the AppSpace folder, or a temporary location, and rename the file as `config.ini`.
   b. Edit the ActiveMatrix BusinessWorks™ Engine Persistence Mode property, `bw.engine.persistenceMode`, and set it to ftgroup.

Follow these steps to configure the engine for group persistence mode, and to set TIBCO Enterprise Message Service™ (EMS) as the group provider technology.

```
bw.engine.persistenceMode=ftgroup
```

c. Specify the group name and group provider technology in the `config.ini` file. The group name is mandatory.

```
# ------------------------------------------------------------
------------------
# Section:  BW Engine Group Configuration.
#
# The properties in this section are applicable to the BW
Engine group.
# Some of the properties in this section are required when the
BW Engine
# property "bw.engine.persistenceMode" is set to "group" or
"ftgroup".
# ------------------------------------------------------------
------------------
# BW Engine Group Name.  This is a required property and it
specifies name of
# the BW engine group.  If this property is not specified,
then the group name
# defaults to "Group_<DomainName>_<AppSpaceName>".
#bw.engine.groupName=mytestgroup

# BW Engine Group Connection Provider Technology.  This is a
required property
# when the persistenceMode is set to "group"
(bw.engine.persistenceMode=group) or
# "ftgroup" (bw.engine.persistenceMode=ftgroup) and it
specifies the BW Engine group
# communication technology. The only supported values are
"ems" and "ftl".
# The group connection provider technology property requires
additional configuration.
# See section "Configuring the Engine for Group Persistence
Mode"
# for additional configuration.
bw.engine.groupProvider.technology=ems
```

d. Specify the group provider configuration:

```
# ------------------------------------------------------------
-----------------
# Section:  BW Engine Group Connection Provider EMS
Configuration.
#
# Some of the properties in this section are required when the
BW Engine Group
# Connection Provider Technology property
"bw.engine.groupProvider.technology"
# value is set to "ems".
# ------------------------------------------------------------
-----------------
# BW Engine Group Connection Provider EMS URL.  This property
is required if
# the group provider technology is "ems".
bw.engine.groupProvider.qin.EMSServerUrl=tcp://localhost:7222

# BW Engine Group Connection Provider EMS User Name.  This
property is required
# if the group provider technology is "ems".
bw.engine.groupProvider.qin.EMSUserName=admin

# BW Engine Group Connection Provider EMS User Password.  This
property is
# required if the group provider technology is "ems".
bw.engine.groupProvider.qin.EMSPassword=

# BW Engine Group Connection Provider EMS Member Prefix.  This
property is
# optional and the default value is "EMSGMS".
#bw.engine.groupProvider.qin.EMSPrefix=EMSGMS

# BW Engine Group Connection Provider EMS Recovery Timeout in
ms.  This
# property is optional and the default value is "5000" ms.
#bw.engine.groupProvider.qin.EMSRecoveryTimeout=5000

# BW Engine Group Connection Provider EMS Recovery Attempt
Delay in ms.  This
# property is optional and the default value is "500" ms.
#bw.engine.groupProvider.qin.EMSRecoveryAttemptDelay=500
```

```
# BW Engine Group Connection Provider EMS Recovery
AttemptCount.  This
# property is optional.
#bw.engine.groupProvider.qin.EMSRecoveryAttemptCount=

# BW Engine Group Connection Provider EMS Connect Attempt
Count.  This property
# is optional.
#bw.engine.groupProvider.qin.EMSConnectAttemptCount=

# BW Engine Group Connection Provider EMS Connect Attempt
Delay in ms.  This
# property is optional.
#bw.engine.groupProvider.qin.EMSConnectAttemptDelay=
```

When setting the password property (`bw.engine.groupProvider.qin.EMSPassword`), the default format is plain text. Run the command `bwadmin obfuscate`, or the command `bwobfuscator`, from the command line to encrypt the password; use the generated encrypted text as the password.

2. **Optional**. The following properties are available for EMS SSL configuration.

```
EMS SSL Configuration
#client identity consisting of the certificate,
#private key and optionally extra issuer certificates can be
included into a single data block using PKCS12.
#Keystore or Entrust Store encodings
#bw.engine.groupProvider.ems.ssl.trust.identity=

#The set of Trusted Certificates represents all trusted issuers of
the server certificate.
#It must be specified by the client application unless the host
certificate verification is completely disabled.
#bw.engine.groupProvider.ems.ssl.trust.certlocation=

#EMS SSL connection trust password. This
#property is required if the JMS server protocol is ssl. The
password may
#be clear text or supplied as an obfuscated string.
#bw.engine.groupProvider.ems.ssl.trust.password=

#trusted certificate commonname must match the ems server hostname
```

```
if set to false
#bw.engine.groupProvider.ems.ssl.disable.verifyHostName=

#client and server certificates must match if set to false
#bw.engine.groupProvider.ems.ssl.trust.disable.verifyHost=
```

3. **Optional.** If you have saved the `config.ini` file to a temporary location, ensure you copy it to the AppSpace root folder in *BW_HOME*`/domains/defaultdomain/appspaces/defaultappspace`.
4. Use the `config` admin command to push the configuration to the AppSpace: `bwadmin [admin] > config -d myDomain -a myAppSpace -cf <`*temporaryLocation*`>/config.ini`.

Engine and Job Tuning

The engine, job tuning, and checkpointing properties are specified in the `config.ini` file for each AppNode and alternatively, at the AppSpace level. The properties specified in the AppSpace `config.ini` file apply to all AppNodes associated with the AppSpace. However the properties specified in the AppNode `config.ini` file only apply to a specific AppNode and furthermore, they overwrite any property specified in the AppSpace `config.ini` file.

The BWEngine is a multi-threaded engine. When events that trigger the execution of a process occur concurrently, the engine runs the same process multiple times, concurrently, once for each event. Each process execution, referred to as a *process instance*, provides an execution scope for the activities that are a part of the process.

Execution of a component process is called a *job*. When the business logic is spread across multiple processes, a process instance is created for each of these processes and run with a particular event. Even though these are separate process instances that they are all working together and can be run as part of the same job.

A job can spawn multiple process instances and can provide the execution context for activities that are part of multiple processes. The engine always runs a job in one engine thread. However, it is not guaranteed that the same engine thread is used for the entirety of the job.

Engine Tuning

The rate at which the BWEngine can run and complete processes depends on the `ThreadCount` and `StepCount` engine properties.

Property	Description
Thread count (`bw.engine.threadCount`)	The process instances in memory are run by the engine. The number of process instances that can be run concurrently by the engine is limited by the maximum number of threads, indicated by the `ThreadCount` property. This property specifies the size of the job thread pool and is applied to all the AppNodes in the AppSpace. Threads run a finite number of

Property	Description
	tasks or activities uninterrupted and then yield to the next process instance that is ready. Engine threads are shared by all the applications deployed on the same AppNode. The CPU and memory resources should be measured under a typical processing load to determine if the default `ThreadCount` is suitable for your environment. By default, the thread count is eight. For instructions on how to change the default value, see Setting Engine and Job Tuning Properties. If the engine throughput has reached a plateau, but the CPU and memory are not fully used, you can increase the thread count to have a positive effect on the throughput. **Caution:** If the engine thread count value is too high, it can cause CPU thrashing, or an increase in latency caused by a large number of messages in the queue. If the engine thread count value is too low, it can cause higher memory use and lower engine throughput as some CPU resources remain unutilized. The process instances created by the engine are typically held in memory. However, this may not be the case if the `FlowLimit` and `PageThreshold` application properties are set. The number of process instances that can be created in memory is also limited by the memory available on the machine and the memory allocated to the JVM on which the engine runs.
Step count `(bw.engine.stepCount)`	The engine `StepCount` property determines the number of activities that are run by an engine thread, without any interruption, before yielding the engine thread to another job that is ready in the job pool. This value is applied to all the AppNodes in the AppSpace. Exceptions to `StepCount` can occur when the job is in a transaction, is blocked, or is waiting for an asynchronous activity to complete.

Property	Description
	When a job is in a transaction, the thread is not released until the transaction is complete, even when the `StepCount` is exceeded. However, if a job is blocked or waiting for an asynchronous activity to complete, the thread can be yielded even when the `StepCount` has not been reached. The default value of this property is `-1`. When the values are set to `-1`, the engine can determine the necessary `StepCount` value. A low `StepCount` value can degrade engine performance due to frequent thread switches. A high `StepCount` value may cause less concurrence in running jobs and hence, result in an inefficient use of CPU.

Job Tuning

Job tuning is done at the application level. Tuning can be narrowed to a specific application version, and a specific component within the application. Job tuning is set by the `FlowLimit`, `PageThreshold`, and `Priority` application properties. When setting these properties, specify the application name. The application version and component name are optional. If the version or component name is not specified, then the property value applies to all versions or components in the application. To push `FlowLimit`, `PageThreshold`, or `Priority` properties to runtime, stop the application, update the property in the AppNode or AppSpace `config.ini` file, and restart the application.

Property	Description
Flow limit `bw.application.job.flowlimit`	The Flow limit property specifies the Flow limit value for an application's process starters or service bindings and is applicable to all the AppNodes in an AppSpace. Flow limit is useful when the engine needs to be throttled as it specifies the maximum number of jobs that can be started before suspending the process starter. Thus ensuring that the incoming requests do not overwhelm the engine performance and the CPU and

Property	Description
	memory is preserved. If the number of jobs being created exceeds the Flow limit, the engine suspends the creation of new jobs but continues running the jobs in memory. The engine resumes creating jobs when sufficient resources are available. There is no default Flow limit value, and it is not enforced by the engine unless the Flow limit property is specified for an application. **Note:** Only set the `bw.application.job.flowlimit` property for applications using non-HTTP-based transports, for example, JMS. If applications are using HTTP-based transports, ensure you set the **Max QTP Threads** value of the **HTTP Connector** shared resource to apply the Flow limit. Get the logs for the component states such as Start, Stop, and Resume based on whether the `FlowLimit` is breached or complied by enabling the core runtime logger `com.tibco.bw.core` at INFO level. **Note:** Flow limit updated via the REST API resets in case of any application/appnode/appspace configuration changes.
Page Threshold `(bw.application.job.pageThreshold)`	The page threshold property specifies the job page threshold value for an application's process starters or service bindings and is applicable to all the AppNodes in an AppSpace. It specifies the maximum number of jobs that can concurrently be loaded into memory, thus limiting the number of running jobs in the memory.

Property	Description
	Jobs are paged out of memory on the basis of a paging strategy selected after the page threshold value is reached. The default value of the strategy is PageOnDelete. There is no default page threshold value and it is not enforced by the engine unless the `PageThreshold` property is specified for an application. The page threshold feature requires that the engine persistent mode property (`bw.engine.persistenceMode`) is set to datastore or group. For more information, see Engine Persistence Modes.
Paging Strategy `(bw.application.job.paging.strategy)`	The paging strategy property specifies how paging should take place. It supports two options: PageOnIdle and PageOnDelete. When you set the value of the property to PageOnIdle, and when the `PageThreshold` is reached, jobs that are idle are moved out of memory, and are paged-out to the engine database. The new or old jobs are loaded back into the memory in place of idle jobs when they are created or scheduled. When you set the value of the property to PageOnDelete, all new jobs created after the `PageThreshold` value is reached are temporarily paged out to the engine database. These jobs are moved back into the memory when the number of jobs in the memory is less than the `PageThreshold` value.
Priority `(bw.application.job.priority)`	The priority property specifies the application job's priority. The option for this property is one of low, normal, or high.

Property	Description
	The default value is normal. Engine threads process lower priority jobs only when higher priority jobs are all blocked from continuing. Lower priority jobs are not preempted while in execution.

You can update the Flow limit value dynamically without restarting an application from the Admin UI. Additionally, you can use the following REST API:

Base path for all REST APIs exposed is `http://<host or IP address>:<port>/` where the port is of running AppNode.

bw/app.json/updateflowlimit/

Method	POST
Description	Update the Flow limit without restarting an application.
Path Parameters	None
Query Parameters	• `parameter: flowLimit` • `Type: Integer` (Mandatory) • `Description:` The new value of the Flow limit. • `parameter: name` • `Type: String` • `Description: Application name`. This property is mandatory for BW 6.x applications but it is optional when using for BWCE or TCI applications. • `parameter: version` • `Type: Integer` • `Description: Application version`. This property is mandatory for BW 6.x applications but it is optional when using for BWCE or TCI

	applications. • `parameter: component` • `Type: String` (Optional) • `Description: Component name of an application.`
For example	`http://<host or IP address>:<port>/bw/app.json/updateflowlimit?flowLimit=<new_flow_limit>&name=<app_name>.application&version=<app_version>&component=<component_name>`

bw/app.json/flowlimit/

Method	GET
Description	Get the latest Flow limit applied to the application or the component without restarting an application.
Path Parameters	None
Query Parameters	• `parameter: name` • `Type: String` • `Description: Application name`. This property is mandatory for BW 6.x applications but it is optional when using for BWCE or TCI applications. • `parameter: version` • `Type: Integer` • `Description: Application version`. This property is mandatory for BW 6.x applications but it is optional when using for BWCE or TCI applications. • `parameter: component` • `Type: String` (Optional) • `Description: Component name of an application.`
For example	`http://<host or IP address>:<port>/bw/app.json/flowlimit?name=<app_name>.application&version=<app_version>&component=<component_name>`

Checkpointing

Property	Description
Retain faulted job `bw.application.checkpoint.retainFaultedJob`	This is an optional property and specifies whether to enable a failed process recovery. The supported values are `true` and `false`. The default value is `false`. The **Application Name** must be included as a part of this property, however, the **Application Version** is optional. After setting these properties in the AppNode `config.ini` file, they can be modified and pushed to runtime by restarting the application. When the property `bw.application.checkpoint.retainFaultedJob` is set to true for an application, the job is not automatically removed after a failure. So the duplicate key remains as long as the job remains. Such a job can be restarted or purged later.
Recover on restart `bw.application.checkpoint.recoverOnRestart`	This is an optional property and specifies whether the checkpointed process instances should automatically restart when a process engine restarts. The supported values are `true` and `false`. The default value is `true`. The **Application Name** must be included as part of this property, and the **Application Version** is optional.
Duplicate key timeout `bw.application.checkpoint.dupKeyTimeout`	This is an optional property and specifies the amount of time in minutes to keep duplicate keys stored after the checkpointed job finished running. By default, the timeout is 0 minutes, and indicates that the duplicate key is deleted as soon as the checkpointed job completes execution. The ActiveMatrix BusinessWorks application name must be included as part of this property. However the application version and component name are

Property	Description
	optional. If the component name is not specified, the value is applied to all components in the ActiveMatrix BusinessWorks application. `bw.application.checkpoint.dupKeyTimeout.<UsersBWApplicationName>[.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=0`
`bw.engine.checkpoint.expired.dupkey.purge.interval`	You can also configure the periodic interval in which expired duplicate keys should be purged from the database by configuring the `bw.engine.checkpoint.expired.dupkey.purge.interval` property. It specifies the default interval for the background thread to poll for expired duplicate keys. The numerical value can be preceded either by "P" or "D". A value "P60" indicates the background thread polls after every 60 minutes whereas a value "D2" indicates the background thread polls after every 2 days. The default value is "P30".

Setting Engine and Job Tuning Properties

Engine and job tuning properties are specified at the **Application Level 2 > Config > Job Tuning** from Admin UI and are set in the Appspace `config.ini` file. These properties are configured from the BWAdmin command line and from the Admin UI at the AppNode level in the AppNode's `config.ini` file. These properties can also be specified at the AppSpace level, but the AppNode property setting takes precedence, and the App Instances go out of sync. However, the AppNode remains in a sync. If you want to apply engine and job tuning property to an application having component in a shared module, use the component name displayed in Admin UI or on CLI.

BWAdmin Command Line

Run the following commands at the command line to set the engine and job tuning properties at the AppNode level or the AppSpace level.

AppSpace Level

1. Copy the existing AppSpace config.ini file (in the root of the AppSpace folder), or the AppSpace config.ini template file appspace_config.ini_template (in BW_HOME/config/) to a temporary location.
2. Uncomment the engine ThreadCount property bw.engine.threadCount and change the default value as needed.

   ```
   bw.engine.threadCount=8
   ```

3. Uncomment the engine StepCount property bw.engine.stepCount and change the default value as needed. The default value of -1 allows the engine to determine the necessary value.

   ```
   bw.engine.stepCount=-1
   ```

4. Uncomment the application Flowlimit property bw.application.job.flowlimit and change the default value as needed. Provide the application name. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the Flowlimit value is applied to all the components in the application.

   ```
   bw.application.job.flowlimit.<UsersBWApplicationName>
   [.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=8
   ```

5. Uncomment the application PageThreshold property bw.application.job.pageThreshold and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the PageThreshold value is applied to all the components in the application.

   ```
   bw.application.job.pageThreshold.<UsersBWApplicationName>
   [.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=10
   ```

6. Uncomment the application paging strategy property bw.application.job.paging.strategy and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the PageThreshold value is applied to all the components in the application.

```
bw.application.job.paging.strategy.<UsersBWApplicationName>
[.<UsersBWApplicationVersion>]
[.<UsersBWComponentName>]=PageOnIdle
```

7. Uncomment the application Priority property bw.application.job.priority and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the Priority value is applied to all the components in the application.

```
bw.application.job.priority.<UsersBWApplicationName>
[.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=normal
```

8. Uncomment the application Retain Faulted Job property `bw.application.checkpoint.retainFaultedJob` and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the Retain Faulted Job value is applied to all the components in the application.

```
bw.application.checkpoint.retainFaultedJob.<UsersBWApplicationNam
e>[.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=false
```

9. Uncomment the application Recover On Restart property `bw.application.checkpoint.recoverOnRestart` and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the Recover On Restart value is applied to all the components in the application.

```
bw.application.checkpoint.recoverOnRestart.<UsersBWApplicationNam
e>[.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=true
```

10. . Save the edited files and use the `config admin` command to push the configuration to the AppSpace:

```
bwadmin[admin] > config -d myDomain -a myAppSpace -cf
<temporaryLocation>/config.ini
```

AppNode level

Procedure

1. Copy the existing AppNode config.ini file (in the root of the AppNode folder) to a temporary location.
2. Uncomment the engine ThreadCount property `bw.engine.threadCount` and change the default value as needed.

   ```
   bw.engine.threadCount=8
   ```

3. Uncomment the engine StepCount property `bw.engine.stepCount` and change the default value as needed. The default value of -1 allows the engine to determine the necessary value.

   ```
   bw.engine.stepCount=-1
   ```

4. Uncomment the application Flowlimit property `bw.application.job.flowlimit` and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the Flowlimit value is applied to all the components in the application.

   ```
   bw.application.job.flowlimit.<UsersBWApplicationName>
   [.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=8
   ```

5. Uncomment the application PageThreshold property `bw.application.job.pageThreshold` and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the PageThreshold value is applied to all the components in the application.

   ```
   bw.application.job.pageThreshold.<UsersBWApplicationName>
   [.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=10
   ```

6. Uncomment the application paging strategy property `bw.application.job.paging.strategy` and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the PageThreshold value is applied to all the components in the application.

```
bw.application.job.paging.strategy.<UsersBWApplicationName>
[.<UsersBWApplicationVersion>]
[.<UsersBWComponentName>]=PageOnIdle
```

7. Uncomment the application Priority property `bw.application.job.priority` and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the Priority value is applied to all the components in the application.

```
bw.application.job.priority.<UsersBWApplicationName>
[.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=normal
```

8. Uncomment the application Retain Faulted Job property `bw.application.checkpoint.retainFaultedJob` and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the Retain Faulted Job value is applied to all the components in the application.

```
bw.application.checkpoint.retainFaultedJob.<UsersBWApplicationNam
e>[.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=false
```

9. Uncomment the application Recover On Restart property `bw.application.checkpoint.recoverOnRestart` and change the default value as needed. The application name must be included in the property. The application version and component name are optional. If the component name is not specified, the Recover On Restart value is applied to all the components in the application.

```
bw.application.checkpoint.recoverOnRestart.<UsersBWApplicationNam
e>[.<UsersBWApplicationVersion>][.<UsersBWComponentName>]=true
```

10. . Save the edited files and use the `config admin` command to push the configuration to the AppNode:

```
bwadmin[admin]> config -d myDomain -a myAppSpace -n myAppNode -cf
<temporaryLocation>/config.ini
```

Admin UI

The application properties, `pageThreshold`, and `FlowLimit` can be passed dynamically from the Admin UI at the Application, AppNode, and AppSpace level.

- To set the job tuning properties at the Application level, open the **Application** page, and click **Configure**. Then click the **Job Tuning** tab, and edit the `pageThreshold` or the `FlowLimit` property as required.
- To set the job tuning properties at the AppSpaces or the AppNodes level, open the **AppSpaces** or the **AppNodes** page and select the AppSpace or AppNode. Click **Configure** and then click the **General** tab to change the value under the **Current Value** column of the required property.

> **Note:** To delete the `pageThreshold` and `FlowLimit` properties, set the property value to zero.

Viewing Engine Properties

View engine properties, including name, step count, thread count, persistence mode, and engine state, from the BWAdmin console.

To view properties, the engine must be running on an AppNode. Open the BWAdmin console and enter the following command to view properties for the engine running on AppNode AN2 in Domain D2 and AppSpace AS2:

```
bwadmin show -d D2 -a AS2 -n AN2 bwengine
```

The following details are displayed:

Property	Description	More Information
Engine Name	The name of the engine running on the specified AppNode.	N/A
Engine Step Count	The number of activities run by an engine thread uninterruptedly before yielding the thread to another job.	Engine and Job Tuning
Engine Thread Count	The maximum number of threads that can be run concurrently.	Engine and Job Tuning
Engine Persistence Mode	The type of collaboration between machines.	Engine Persistence Modes
Engine State	The state of the engine.	N/A

Engine Properties

ActiveMatrix BusinessWorks can be used to set engine properties at the AppNode or AppSpace `config.ini` file. Configure the engine by changing or assigning appropriate values to the properties. For more information, see `appspace_config.ini_template` or `appnode_config.ini_template` files at the `{TIBCO_HOME}\bw\6.x\config` directory.

BWEngine General Configuration

The properties in this section are applicable to the BWEngine.

Property	Description
`bw.engine.threadCount`	Engine thread count. It specifies the engine thread pool size. Value of this property must always be greater than 0.

Property	Description
	The default value is 8.
bw.engine.stepCount	Engine step count. It specifies the number of activities to run for a process instance, before the BWEngine yields the thread. The default value is -1.
bw.engine.separate.logs.by.app	Application level logging. It enables separate log files at the application level. The default value is false.
bw.engine.node.weight	Set the value between 1 to 99. The AppNode with highest weight is the primary node.
bw.engine.name	Specify the name of the engine. The default value is Main.
bw.engine.persistenceMode	Specifies engine execution mode. The default value is memory.
bw.engine.shutdownOnFailure	BWEngine Shut down Option. The default value is true.

Property	Description
bw.engine.activity.async.waitTime	BW Asynchronous Activity Timeout. It specifies the default timeout or wait time value in milliseconds for the asynchronous activities run by the BWEngine. The default value is 180000 (3 minutes).
bw.engine.activity.signalin.eventTimeout.purge.interval	Specifies the default interval for the background thread that looks for expired messages. The value must be specified in minutes. The default value is 30 minutes.
bw.engine.show.all.errors.while.application.startup	Lists all errors within an application during an application startup. The default value is false. This is a two-phase process: 1. **Deserialization phase**: In this phase, all deserialization errors of processes are shown in the logs.

Property	Description
	Once all of the deserialization errors are resolved, restart an application to see initialization errors. 2. **Initialization phase**: In this phase, all initialization errors related to activities are shown in the logs.
`bw.engine.checkpoint.expired.dupkey.purge.interval`	It specifies the default interval for the background thread to poll for expired duplicate keys. The default value is "P30".
`bw.engine.inline.subprocess.multiLogging.disable`	Set the property to true if you do not want to list ERRORs generated by the inline subprocess on the console or in the AppNode's log files.
`bw.engine.enable.memory.saving.mode`	This property, when set to true, enables usage of memory saving mode, which frees activity output variables once they are no longer

Property	Description
	needed. The default value is true. To disable the memory saving mode, unselect the **Save information to support memory saving mode** checkbox available at **Window > Preferences > BusinessWorks > Process Diagram** in the Memory Saving Mode section. Then, to remove the memory saving variable, right-click **ActiveMatrix BusinessWorks™ Projects** and select **Refactor > Repair BusinessWorks Projects**. In the dialog, select the **Remove memory saving variables** option. On clicking the **Preview** button, the variables that can be removed from different activities are displayed on the Preview page, then click **OK**. To update existing projects, use the repair tool.

BWEngine Database Configuration

The properties in this section are applicable to the BWEngine database. All properties in this section are mandatory, when the BWEngine property `bw.engine.persistenceMode` is

set to datastore or group.

Property	Description
bw.engine.db.jdbcDriver	The BWEngine database driver.
bw.engine.db.url	The BWEngine database URL.
bw.engine.db.userName	The BWEngine database username.
bw.engine.db.password	The BWEngine database password.
bw.engine.db.maxConnections	The number of connections that can be made to the BWEngine database.

BWEngine Group Configuration

The properties in this section are applicable to the BWEngine group. Some of the properties in this section are mandatory when the BWEngine property bw.engine.persistenceMode is set to group or ftgroup.

Property	Description
bw.engine.groupName	It specifies the name of the BWEngine group. If this property is not specified, then the group name defaults to Group_<DomainName>_ <AppSpaceName>"
bw.engine.group.ats.timeout	BWEngine Active to Standby Timeout property specifies the time to wait (in seconds) before force stopping an application on an AppNode that is transitioning from active to standby state. The default value is 60 secs. If the value is set to 0, it indicates that the AppNode waits till the application is gracefully stopped.
bw.engine.groupProvider.technology	BWEngine Group Connection Provider Technology. This is a required property when the

Property	Description
	`bw.engine.persistenceMode` property is set to group or ftgroup. The supported values for the `bw.engine.groupProvider.technology` property are `ems` and `ftl`.

BWEngine ftgroup Properties

Property	Description
`bw.engine.ftgroup.lbmode`	When this property is false, all processes run on a single engine in the group. One of the engines in the group takes over in the event that the primary engine fails. When this property is true, all Multiple AppNodes processes run on all the engines in the group. This property only applies when the BWEngine property `bw.engine.persistenceMode` is set to ftgroup. The default value is false.
`bw.engine.use.weighted.node`	Indicates whether node weights should be used. The property is applied at the AppSpace level. When the property is set to true, the node with the highest weight is chosen as the primary node in the group. Use the property `bw.engine.node.weight` in the node config to specify that node's weight. This property only applies when the BWEngine property `bw.engine.persistenceMode` is set to ftgroup. When you want to elect an AppNode as a leader AppNode, then set the `bw.engine.use.weighted.node` property to true at an AppSpace level.

Property	Description
	The default value is false.

BW Event Configuration Properties

The properties in this section are applicable to the BW Event Publisher and various BW Event Subscribers that consume the generated events.

Property	Description
`bw.engine.event.publisher.enabled`	Enable or disable the BWEngine Event Publisher property specifies whether BWEngine Event Publisher should be enabled or disabled in the BWEngine. The default value is true.

BWEngine Group Connection Provider EMS Configuration

Some of the properties in this section are mandatory when the BWEngine Group Connection Provider Technology property `bw.engine.groupProvider.technology` value is set to `ems`.

Property	Description
`bw.engine.groupProvider.qin.EMSServerUrl`	Mandatory. The BWEngine Group Connection Provider EMS URL.
`bw.engine.groupProvider.qin.EMSUserName`	Mandatory. The BWEngine Group Connection Provider EMS Username.
`bw.engine.groupProvider.qin.EMSPassword`	Mandatory. The

Property	Description
	BWEngine Group Connection Provider EMS User Password.
bw.engine.groupProvider.qin.EMSPrefix	The BWEngine Group Connection Provider EMS Member Prefix. The default value is EMSGMS.
bw.engine.groupProvider.qin.EMSRecoveryTimeout	The BWEngine Group Connection Provider EMS Recovery Timeout in milliseconds. The default value is 5000.
bw.engine.groupProvider.qin.EMSRecoveryAttemptDelay	The BWEngine Group Connection Provider EMS Recovery Attempt Delay in milliseconds. The default value is 500.
bw.engine.groupProvider.qin.EMSRecoveryAttemptCount	The BWEngine Group Connection Provider EMS Recovery AttemptCount.
bw.engine.groupProvider.qin.EMSConnectAttemptCount	The BWEngine Group Connection Provider EMS Connect Attempt Count.
bw.engine.groupProvider.qin.EMSConnectAttemptDelay	The BWEngine Group Connection Provider EMS Connect Attempt

Property	Description
	Delay in milliseconds.
bw.engine.groupProvider.ems.ssl.trust.identity	EMS SSL configuration client identity consisting of the certificate, private key, and optionally extra issuer certificates can be included into a single data block using PKCS12, Keystore, or Entrust Store encodings.
bw.engine.groupProvider.ems.ssl.trust.certLocation	The set of Trusted Certificates represents all trusted issuers of the server certificate.
bw.engine.groupProvider.ems.ssl.trust.password	EMS SSL connection trust password.
bw.engine.groupProvider.ems.ssl.disable.verifyHostName	Trusted certificate common name must match the EMS server hostname if set to false.
bw.engine.groupProvider.ems.ssl.trust.disable.verifyHost	The client and server certificates must match if set to false.

BWEngine Group Connection Provider FTL Configuration

Some of the properties in this section are mandatory when the BWEngine Group Connection Provider Technology property `bw.engine.groupProvider.technology` value is set to `ftl`.

Property	Description
bw.engine.groupProvider.ftl.realmserver	Mandatory. BWEngine Group Connection Provider FTL Realm Server. The default value is http://localhost:8080
bw.engine.groupProvider.ftl.username	Mandatory. The BWEngine Group Connection Provider FTL Realm client username.
bw.engine.groupProvider.ftl.password	Mandatory. The bwngine Group Connection Provider FTL Realm client password.
bw.engine.groupProvider.ftl.appinstance.id	Mandatory. The BWEngine Group Connection Provider FTL application identifier.
bw.engine.groupProvider.ftl.secondaryserver	The BWEngine Group Connection Provider FTL secondary realm server.
bw.engine.groupProvider.ftl.groupname	Mandatory. The BWEngine Group Connection Provider FTL group name.
bw.engine.groupProvider.ftl.appname	Mandatory. The BWEngine Group Connection Provider FTL application name.
bw.engine.groupProvider.ftl.publish.endpoint	Mandatory. The BWEngine Group Connection Provider FTL publish endpoint.
bw.engine.groupProvider.ftl.subscribe.endpoint	Mandatory. The BWEngine Group Connection Provider FTL application name.

Property	Description
`bw.engine.groupProvider.ftl.client.retries`	Use the property to set the FTL property `TIB_REALM_PROPERTY_LONG_CONNECT_RETRIES`. The default value is 5. To retry forever, set the value to 0. If the connect call cannot connect to the FTL server after the maximum number of connection attempts, an exception is displayed.

Governance and Monitoring

Enable the agents in the AppNode to monitor applications, to enforce policies, to view statistics, and monitor process instances for an environment in ActiveMatrix BusinessWorks.

Monitoring Processes

Using the process monitoring feature you can observe and check the status of process instances from the Admin UI.

All the process instances in the application are grouped by packages, and you can monitor the status of the process instances and subprocesses that were successfully run, canceled or faulted.

Details such as input data, output data, fault data and other configuration details for the activities are also available by viewing the process diagram for the instances.

Enabling Process Monitoring

Process monitoring can be configured from the `bwagent_db.json` or `bwagent_ftl.json` files. Set the property `statsprovider` to true to enable process monitoring.

You can use the same or different databases for process monitoring.

Process Monitoring

The transport layers communicate between the BWAgent and the AppNode, and the supported transport layers are REST, UDP, and FTL. The default transport layer is REST. For the REST transport layer, the default monitor data format is JSON. This is the default setting in the BWAgent and the AppNode.

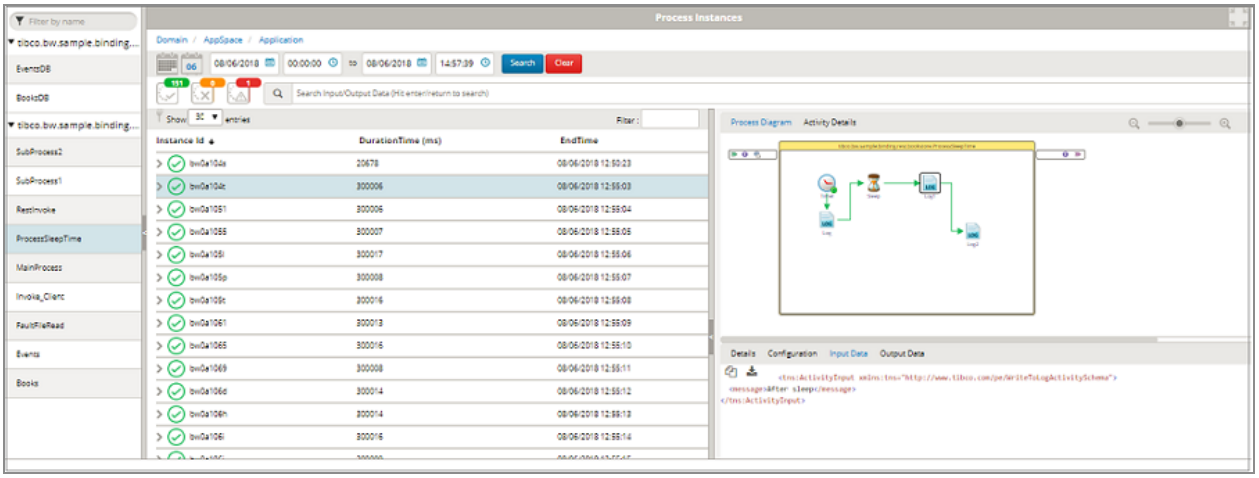
The `statstransport` property is also added to the AppNode configuration file when the AppNode is created.

For more information about REST, UDP, and FTL configurations, see Configuring using REST, Configuring using UDP and Configuring using FTL respectively.

To access the process monitoring landing page, go to the **Application Level 2** page, navigate to the **Process** tab, click the **Graph View** icon and click the **Process Instance** icon. You can also use the shortcut key Shift + P to navigate to the process monitoring landing page directly.

All the instances, processes, and subprocesses of the selected application are displayed on the landing page.

You can begin monitoring your process instances once you enable the Process Instance icon after deploying the application.



By default, all the instances in the selected process are displayed. Subprocesses, if any, can be viewed by expanding the process.

Note: Double-click navigation is supported up to nine subprocesses, and only works for direct subprocesses. Service subprocesses are not supported.

In the above example, click the process Books. Job data related to the Books process is displayed in a tabular form, and the process diagram of the process is also displayed.

Additional Features

In the Admin UI, the following details are displayed in the default view.

- **Instance Id** - displays all the (instance ids of the) process instances.
- **DurationTime (ms)** - displays the total time taken to run the process instance (in milliseconds)
- **EndTime** - the time when the process instance ended.

The columns displayed in the default view can also be customized to display additional information about the process instances. Use the **Select Columns** filter to add the columns, **AppNodes**, **StartTime**, and **DurationTime (ms)**.

The other filters provided in Admin UI are:

Filter	Description
Date filters	Job data can be filtered for a particular date or a time range. Alternatively, the Calendar icon filters and displays job data for Last Month and Today.
Job Status filters	Job data can be filtered based on their completion status. Select the icon to filter the jobs that were completed. The icon displays only the jobs that are canceled and filters the jobs that are faulted.
Search Input/Output Data	This filter searches through the input and output data and displays the required information.
Filter	This filter searches through the instances that are displayed for any specific value provided in this filter text box.

Navigating through the UI

To navigate through the Domain quickly, the Admin UI also provides breadcrumb navigation. The **Domain** link in the breadcrumb navigation, **Domain/ AppSpaces/ Application** navigates to the page where all the applications within the domain are displayed. The **AppSpace** link navigates to the **AppSpace level 2 > Application** tab, and the **Application** link navigates to the page where all the application instances for all the processes are displayed.

The process diagram and activity details for each process instance is displayed in the extreme right panel. Click the process instance in the second panel, and the process diagram for that instance is displayed. The executed transitions and flow are displayed. The **Details** tab, **Configuration**, **Input Data** and **Output Data** tabs contain the configuration, input, and output details of the process instance.

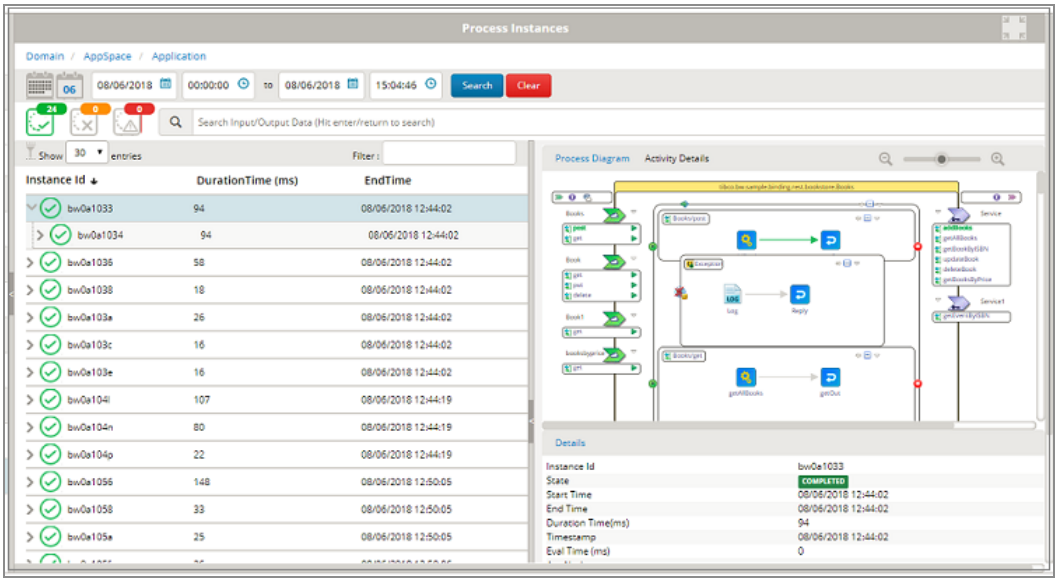
To disable or enable input and output data storage for audit events while using the statistics collection, use the following BWEngine REST APIs:

- To disable:

  `http://<host>:<appnode port>/bwm/monitor.json/disableinputoutputdataforauditevents`

- To enable:

  `http://<host>:<appnode port>/bwm/monitor.json/enableinputoutputdataforauditevents`

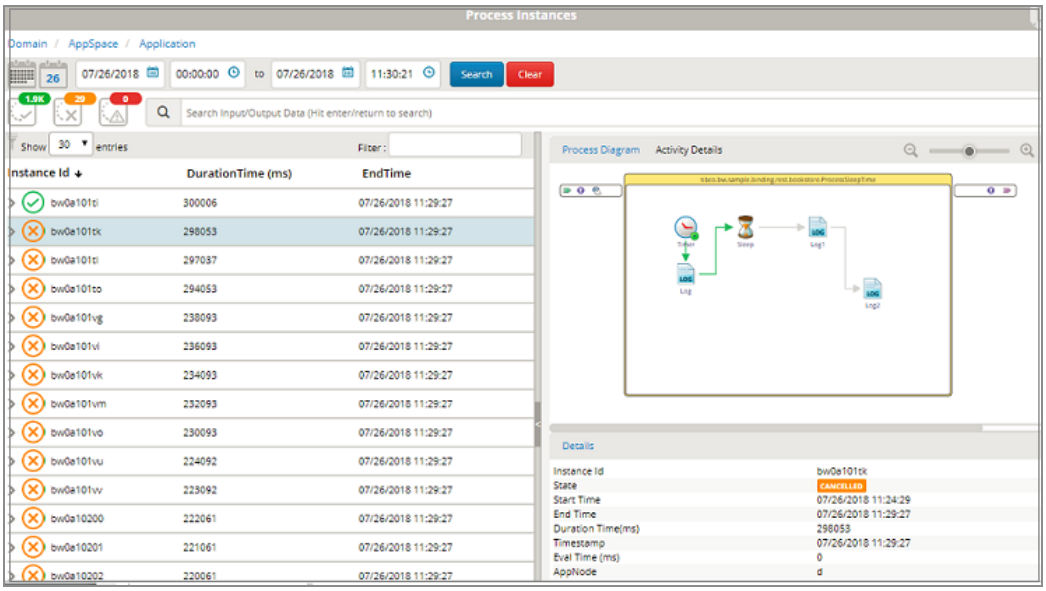


Selecting any instance from the table highlights the reference or service, if any, and also highlights in green, the activity transitions that were successfully run.

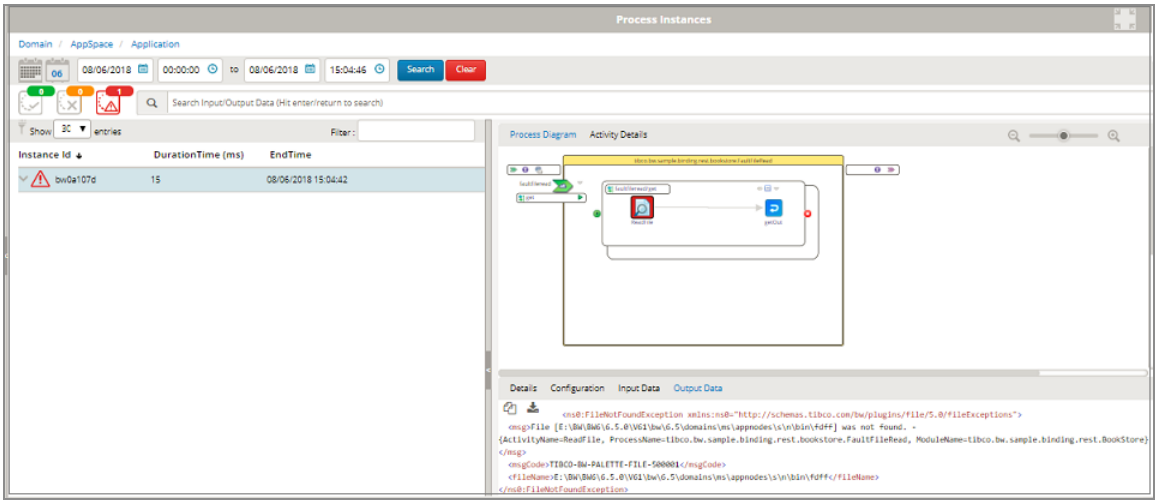
> **Note:** When a process contains multiple constructors, the activities and transitions in the constructor are not visible in the Admin UI when the constructor is minimized while creating the EAR file. Expand the constructors and regenerate the EAR file to view the transitions inside the constructors.

Fix any ActivityID-related warnings displayed in Studio, and then create the EAR file to ensure the plotting and Input and Output data is displayed correctly.

The process monitoring for canceled jobs displays the successful transactions in green up until the point where the process was successfully run.



Processes that are faulted due to errors are highlighted in red. The **Ouput Data** tab displays the error due to which the process has faulted.



Process Monitoring with an HTTPS AppNode

To use process monitoring with an AppNode running in the HTTPS mode, perform the following steps before starting the BWAgent:

Procedure

1. Generate the keystore file with the local host added in the Subject Alternative Name

(SAN).

2. Optional. Extract the `.crt` file from the generated keystore by running the following command:

```
keytool -export -alias <aliasname> -file server.crt -keystore
server1.jks
```

3. Import the extracted `.crt` file in `BW_HOME/tibcojre4/lib/security` by running the following command:

```
keytool -import -alias -<aliasname> -file <path>/server.crt -
keystore cacerts
```

> **Note:** Use `changeit` as the default password for the JRE cacerts file. Also, use a new alias every time.

Configuring using REST

You can configure process monitoring using REST.

Procedure

1. Update the following properties in the BWAgent configuration files `bwagent_db.json` or `bwagent_ftl.json` based on your BWAgent configuration.
   - Set `statstransport` to REST
   - Set `statsdataformat` to json
   - Set `statsprovider` to true
   - Set `dbprovidertype` to *<db type>*
   - Set `dbproviderdriver` to *<db provider>*
   - Set `dbproviderconnectionurl` to *<db connection url>*
   - Set `dbprovideruser` to *<db user>*
   - Set `dbproviderpassword` to *<db password>*

   Run the BWAdmin config command with the `-cf` option to push the changes from

the BWAgent configuration JSON file to the `bwagent.ini` file.

- To start the BWAgent in the `dbems` mode

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

- To start the BWAgent in the dbftl mode

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_ftl.json agent
```

Start the BWAgent and create the AppNode(s).

2. To ensure minimum data loss, set the values of the following two AppNode properties on the lower side: For example, you can set `bw.monitor.publishtimer= 1500` and `bw.monitor.batchsize= 1`.

   batchsize: Process Monitoring data is published in batches. This property specifies batch size for the data.

   publishtimer: This property specifies the time interval for publishing Process Monitoring data.

   The criteria that is fulfilled first is considered first.

   You can configure the AppNode properties using the Configure icon from the Admin UI AppNode level 2 page or from the BWAppNode's `config.ini` file.

   Ensure the monitor provider property (`bw.monitor.provider=REST`) is present in the AppNode `config.ini` file and in the `bwagent.ini` file.

3. Start the AppNodes.
4. Upload and deploy the Application. Start the Application.
5. You can enable process monitoring for any particular application by navigating to the Application Level 2 page, turning the **Process Monitor** button **ON** and restarting the Application.
6. Navigate to **Application Level 2 > Processes > Graph View > Process Instance**. Alternatively you can also use the shortcut key Shift + P from the Application Level 2 Page to open the Process Monitoring landing Page directly.

   To enable process monitoring from CLI, run the following commands:

```
bwadmin[admin]> enablestats -d domain -a appspace processinstance
```

```
b1.application 1.0
TIBCO-BW-ADMIN-300413: Enabled statistics collection for
Application [b1.application:1.0].

bwadmin[admin]> enablestats -d domain -a appspace activityinstance
b1.application 1.0
TIBCO-BW-ADMIN-300413: Enabled statistics collection for
Application [b1.application:1.0].
```

Alternatively, you can run the single command as follows:

```
bwadmin[admin]> enablestats -d domain -a appspace processmonitor
b1.application 1.0
TIBCO-BW-ADMIN-300413: Enabled statistics collection for
Application [b1.application:1.0].
```

7. Select any activity in the process diagram and verify the details in the Details, Configurations, Input data, or the Output data tabs.

   In your production environment, it is recommended to use TIBCO FTL®.

Configuring using UDP

You can configure the process monitoring using UDP.

Procedure

1. Update the following properties in the BWAgent configuration files `bwagent_db.json` or `bwagent_ftl.json` based on your BWAgent configuration.
   - Set `statstransport` to UDP
   - Set `statsprovider` to true
   - Set `dbprovidertype` to *<db type>*
   - Set `dbproviderdriver` to *<db provider>*
   - Set `dbproviderconnectionurl` to *<db connection url>*
   - Set `dbprovideruser` to *<db user>*
   - Set `dbproviderpassword` to *<db password>*

   Run the BWAdmin `config` command with the `-cf` option to push the changes from

the BWAgent configuration JSON file to the `bwagent.ini` file.

- To start the BWAgent in the `dbems` mode

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

- To start the BWAgent in the `dbftl` mode

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_ftl.json agent
```

2. Start the BWAgent and create the AppNode(s).
3. To ensure minimum data loss, set the values of the following two AppNode properties on the lower side: For example, you can set `bw.monitor.publishtimer= 1500` and `bw.monitor.batchsize= 1`.

   `Batchsize`: Process Monitoring data is published in batches. This property specifies the batch size for the data.

   `Publishtimer`: This property specifies the time interval for publishing Process Monitoring data.

   The criteria that is fulfilled first is considered first.

   You can configure the AppNode properties using the Configure icon from the Admin UI AppNode level 2 page or from the BWAppNode's `config.ini` file.

   Ensure the monitor provider property (`bw.monitor.provider=UDP`) is present in the AppNode `config.ini` file and in the `bwagent.ini` file.
4. Start the AppNode(s).
5. Upload and deploy the Application. Start the Application.
6. You can enable process monitoring for any particular application by navigating to the Application Level 2 page, turning the **Process Monitor** button **ON** and restarting the Application.
7. Navigate to **Application Level 2 > Processes > Graph View > Process Instance**. Alternatively you can also use the shortcut key Shift + P from the Application Level 2 Page to open the Process Monitoring landing Page directly.

   To enable process monitoring from the CLI, run the following commands:

```
bwadmin[admin]> enablestats -d domain -a appspace processinstance
b1.application 1.0
TIBCO-BW-ADMIN-300413: Enabled statistics collection for
Application [b1.application:1.0].

bwadmin[admin]> enablestats -d domain -a appspace activityinstance
b1.application 1.0
TIBCO-BW-ADMIN-300413: Enabled statistics collection for
Application [b1.application:1.0].
```

Alternatively, you can run the single command as follows:

```
bwadmin[admin]> enablestats -d domain -a appspace processmonitor
b1.application 1.0
TIBCO-BW-ADMIN-300413: Enabled statistics collection for
Application [b1.application:1.0].
```

8. Select any activity in the process diagram and verify the details in the Details, Configurations, Input data, or the Output data tabs.

   In your production environment, it is recommended to use TIBCO FTL®.

Configuring using FTL

You can configure process monitoring using TIBCO FTL ®.

Procedure

1. Download the latest FTL driver and install TIBCO FTL®. Refer to the TIBCO FTL® *Installation* guide for installation instructions. Set the property in `bwcommon.tra -tibco.env.FTL_HOME=<FTL_HOME>` and install the FTL driver using the `bwinstall` utility.

   The *FTL_HOME* path provided should be till the version folder. For example, `tibco.env.FTL_HOME=/opt/tibco/ftl/5.4`

2. Start the FTL realm server using `./tibrealmserver -ht <IP address>:8080`.
3. Run the FTL command, `-./tibrealmadmin -rs http://<IP address>:8080 -ur <PATH of bwadmin_ftlrealmserver.json>`.

   Two applications are created on the FTL server.

4. Update the following properties in the `bwagent_db.json, bwagent_as.json, or`

`bwagent_ftl.json` file, based on your BWAgent configuration.

- Set `statstransport` to FTL
- Set `statsdataformat` to bytestream
- Set `statsprovider` to `true`
- Set `dbprovidertype` to *<db type>*
- Set `dbproviderdriver` to *<db provider>*
- Set `dbproviderconnectionurl` to *<db connection url>*
- Set `dbprovideruser` to *<db user>*
- Set `dbproviderpassword` to *<db password>*
- Set `statsftlrealmserverurl` to http://*<IP Address>*[:port]

  In case of FTL 6.x server in FT mode, set multiple `realmserver` values separated by pipe. (|).

  For example: `bw.agent.technology.dbftl.ftl.realmserver= http://10.97.240.76:8050 | http://10.97.240.76:8051 | http://10.97.240.76:8052`

  If any of the configuration settings are different from the default settings, update the following additional properties as applicable.

- `statsftlapplicationname`
- `statsftlsecondaryurl`

  This property is only applicable for FTL 5.x. To use this property for FTL 6.x, set the `statsftlsecondary` to true.

  By default, the `statsftlsecondary` property is set to false.

- `statsftlusername`
- `statsftluserpassword`
- `statsftlendpoint`
- `statsftldataformat`
- `statsftlinbox`

Run the BWAdmin config command with the `-cf` option to push the changes from the BWAgent configuration JSON file to the `bwagent.ini` file.

- To start the BWAgent in the `dbems` mode

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_db.json agent
```

- To start the BWAgent in the `dbftl` mode

```
BW_HOME\bin>bwadmin config -cf ../config/bwagent_ftl.json agent
```

5. Start the BWAgent and create the AppNode(s).

   Ensure the monitor data format property (`bw.monitor.data.format=bytestream`) and the monitor provider property (`bw.monitor.provider=FTL`) are present in the AppNode `config.ini` file and in the `bwagent.ini` file.

6. Upload and deploy the Application. Start the Application.

7. You can enable process monitoring for any particular application by navigating to the Application Level 2 page, turning the **Process Monitor** button **ON** and restarting the Application.

   To enable process monitoring from the CLI, run the following commands:

```
bwadmin[admin]> enablestats -d domain -a appspace processinstance
b1.application 1.0
TIBCO-BW-ADMIN-300413: Enabled statistics collection for
Application [b1.application:1.0].

bwadmin[admin]> enablestats -d domain -a appspace activityinstance
b1.application 1.0
TIBCO-BW-ADMIN-300413: Enabled statistics collection for
Application [b1.application:1.0].
```

   Alternatively, you can run the single command as follows:

```
bwadmin[admin]> enablestats -d domain -a appspace processmonitor
b1.application 1.0
TIBCO-BW-ADMIN-300413: Enabled statistics collection for
Application [b1.application:1.0].
```

8. Navigate to **Application Level 2 > Processes > Graph View > Process Instance**.

   Alternatively, you can also use the shortcut key Shift + P from the Application Level 2 Page to open the Process Monitoring landing Page directly.

Configuring with CSV

You can now get activity and process statistics in .csv files.

Procedure

1. Set the following properties in the `AppNode.config` file:
   - `bw.monitor.provider=csv`
   - `bw.monitor.data.format=bytestream`
2. Update the following properties in the `bwagent_db.json` and `bwagent_ftl.json` files:
   - Set `statsprovider` to `true`
   - Set `statsdataformat` to `bytestream`
   - Set `statstransport` to `csv`

   By default, the `statsprovider` property value is false.

   The generated `.csv` file does not contain input and output data. It contains a limited data set as follows:

   `processstats.csv` contains the following information:

   - Application Name
   - Application Version
   - Module Name
   - Module Version
   - Component Process Name
   - JobId
   - Parent Process Name
   - ParentProcessInstanceId
   - Process Name
   - ProcessInstanceId
   - Start Time (Milliseconds)
   - End Time (Milliseconds)

- Elapsed Time (Milliseconds)
- Evaluation Time (Milliseconds)
- Status

`activitystats.csv` contains the following information:

- Application Name
- Application Version
- Module Name
- Module Version
- Activity Name
- Process Name
- ProcessInstanceId
- Start Time (Milliseconds)
- Elapsed Time (Milliseconds)
- Evaluation Time (Milliseconds)
- Status

Application Statistics Collection

You can collect three types of statistics for an application: application job metrics, process statistics, and execution statistics.

For more information, see the following sections:

- Application job metrics - Application Metrics
- Process instrumentation statistics for ActiveMatrix BusinessWorks 6.x processes and activities - Process Statistics
- Execution statistics for ActiveMatrix BusinessWorks 6.x processes and activities - Process Execution Statistics

Application Metrics

Application metrics provides the job statistics of an application. You can view application metrics from the command line, or from the Admin UI.

Command Line

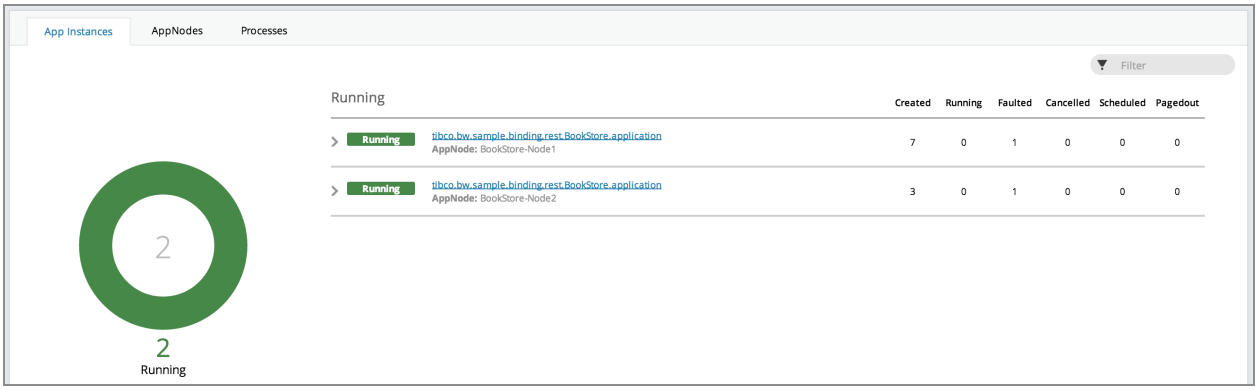
To view application metrics from the command line, run the Admin CLI command `show metrics`.

> **Note:** The application metrics collection functionality is enabled by default, but can be disabled by setting the property, `bw.frwk.event.subscriber.metrics.enabled`, to false in the AppSpace `config.ini` file. Use the BWAdmin `config` command to push this configuration to the AppSpace.

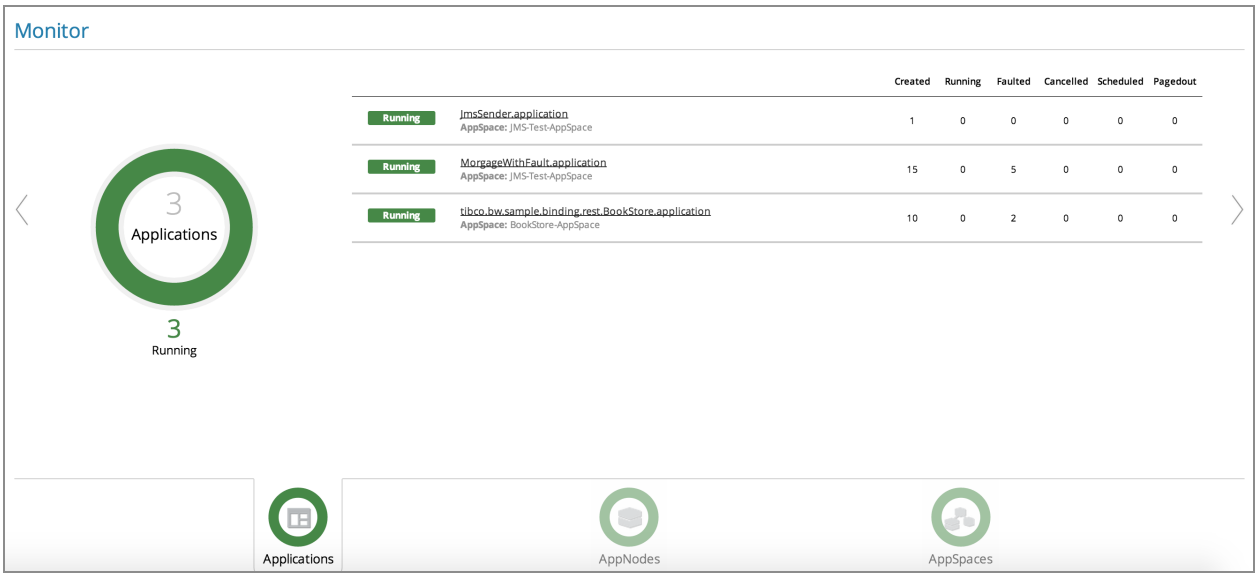
Admin UI

You can view application metrics in the Admin UI from the **Monitoring** tab, or the **Applications** tab.

To view job count statistics for multiple applications, click the **Monitoring** tab.



To view job count statistics for a specific application, click the **Applications** tab, select the application you want to view, and switch to the monitoring view by clicking the icon.



Process Statistics

Process statistics collection can be enabled or disabled from the command line, or from the Admin UI.

Enabling and Disabling Process Statistics

Process statistics can be enabled and disabled from the command line or from the Admin UI.

Command Line

Enable or disable the collection of process statistics for all applications in an AppNode by using the `enablestats` and `disablestats` commands respectively.

> **Note:** To view command syntax, use the help option on the BWAdmin console. For example, enter `help enablestats` or `help disablestats` to view the syntax for the `enablestats` command or the `disablestats` command.

To enable collection of statistical data for all processes running in an AppNode at the time of an application startup, set the `bw.frwk.event.subscriber.instrumentation.enabled` property to TRUE in the AppSpace `config.ini` file.

If you set the property to FALSE the process instrumentation statistics is disabled at the time of an application startup.

If the property is not set, the previous state of the process instrumentation persists.

- **Enable Process Statistics**
  - To enable process statistics for all applications on a single AppNode, run the following command:

    ```
    enablestats -d defaultdomain -a MyAppSpace -n MyAppNode
    process
    ```

  - To enable process statistics for a single application on an AppNode, run the `enablestats` command, and specify the application name and version. In the following example, the application testApp and version 1.0 are provided in the command syntax:

    ```
    enablestats -d defaultdomain -a MyAppSpace process testApp 1.0
    ```

- **Disable Process Statistics**
  - To disable process statistics for all applications on a single AppNode, run the following command:

    ```
    disablestats -d defaultdomain -a MyAppSpace -n MyAppNode
    process
    ```

  - To disable process statistics for a single application on an AppNode, run the `disablestats` command, and specify the application name and version. In the following example, the application testApp and version 1.0 are provided in the command syntax:

    ```
    disablestats -d defaultdomain -a MyAppSpace process testApp
    1.0
    ```

Admin UI

Application statistics collection can be enabled or disabled from the Admin UI by setting the following properties.

Property	Description
Process Instrumentation	Click **ON** to enable process instrumentation data collection. To enable data collection of an application that is running on multiple AppNodes, click the **Application** tab, and enable the **Process Instrumentation** property. Process instrumentation statistics is collected for all application instances. To enable data collection of all applications running on an AppNode, click the **AppNodes** tab, and enable the **Process Instrumentation** property. Process instrumentation statistics is collected for all applications running on the specified AppNode.
Process Monitor	Click **ON** to enable process monitoring to view the process instances. To enable process monitoring of an application that is running on multiple AppNodes, click the **Application** level 2 page, and click **Process Monitor**. Process monitoring is enabled for all application instances. To enable process monitoring of all applications running on an AppNode, click the **AppNodes** tab and click **Process Monitor**. Process monitoring is enabled for all applications running on the specified AppNode.

Viewing Collected Statistics

You can view process statistics through the command line, or the Admin UI.

Before you begin

Ensure that you have enabled process statistics collection. For more information about how to do this, see Enabling and Disabling Process Statistics.

Command Line

You can use the following commands on the OSGi console to view collected statistics:

- bw:lpis to print statistics of processes that have been run for the application.
- bw:lais to retrieve statistics for activities that have been run in processes for the

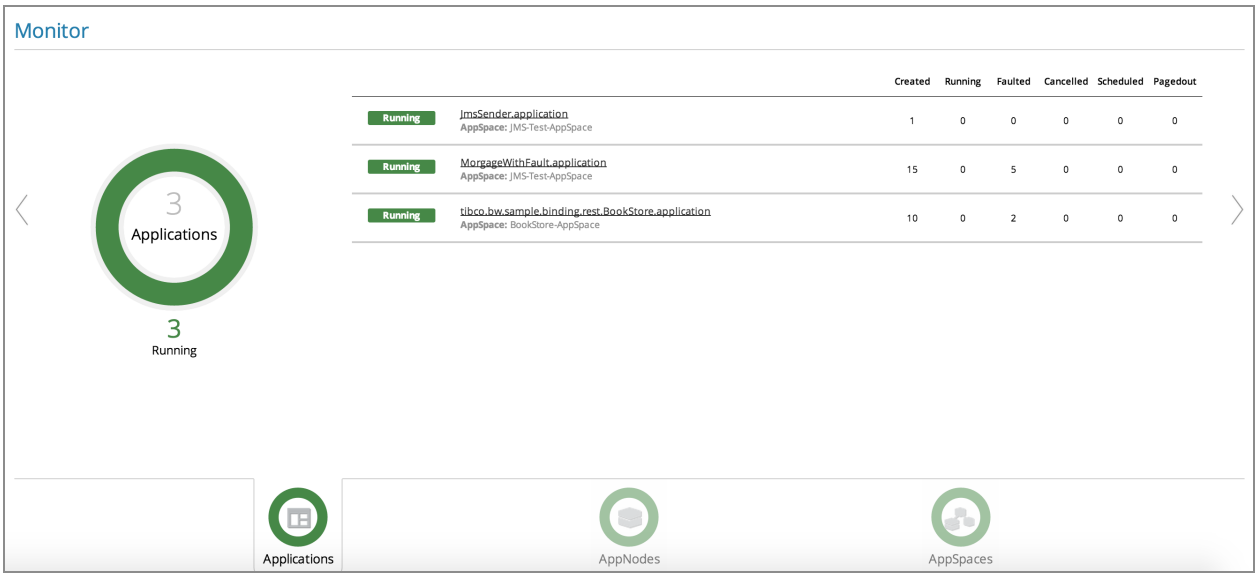
application.

Admin UI

For details about how to view application, process, and activity data from the Admin UI, see the following sections.

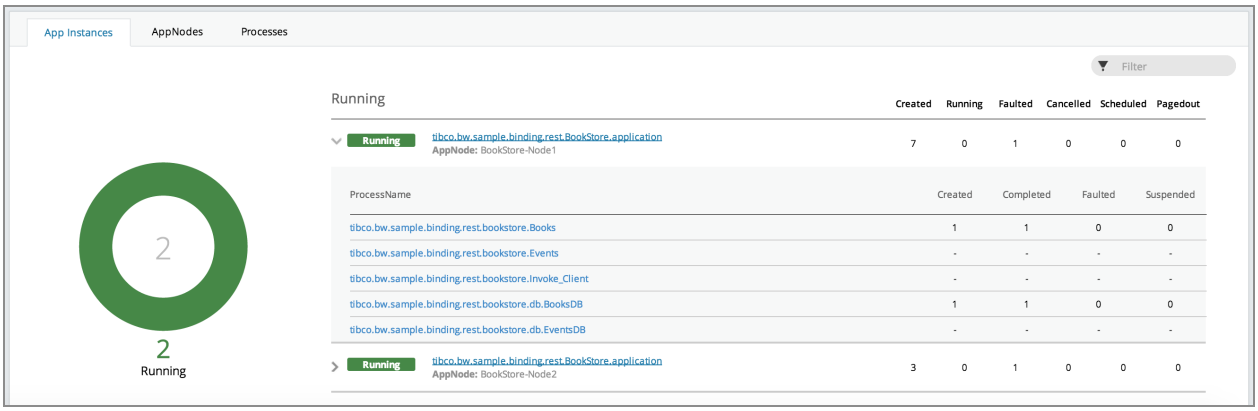
View Application Data

To view application job counts on each AppNode, select the **Application** tab, and click the icon to switch to the Monitoring View.

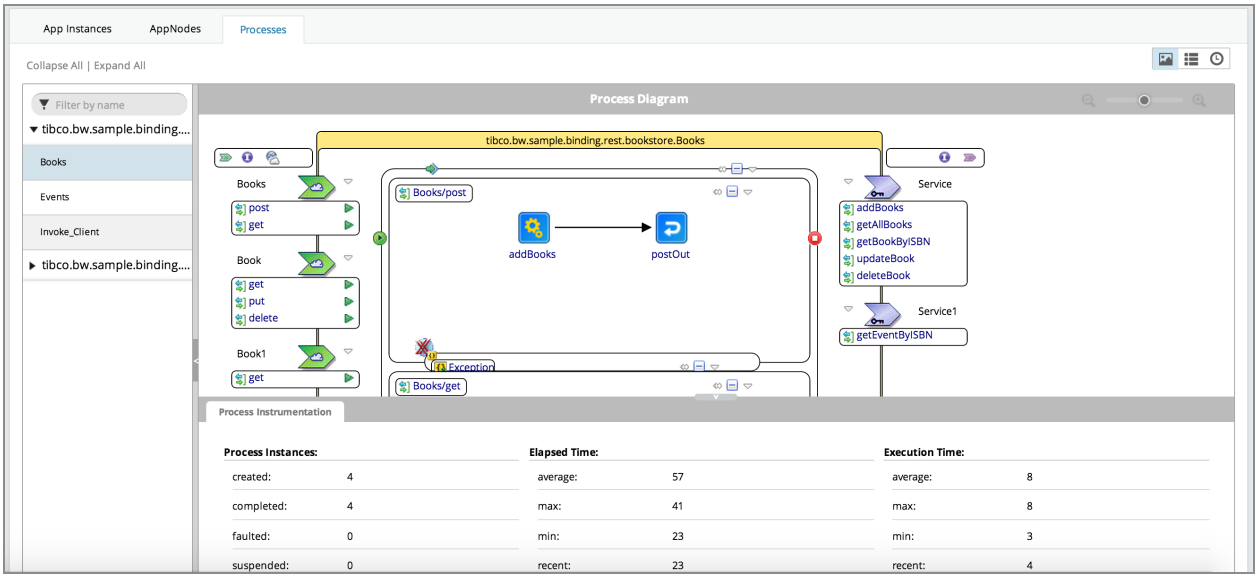


View Process Data

Ensure you are on the **Application** tab, and click the icon to switch to the Monitoring View. Next, select an application and expand it to view job process counts.

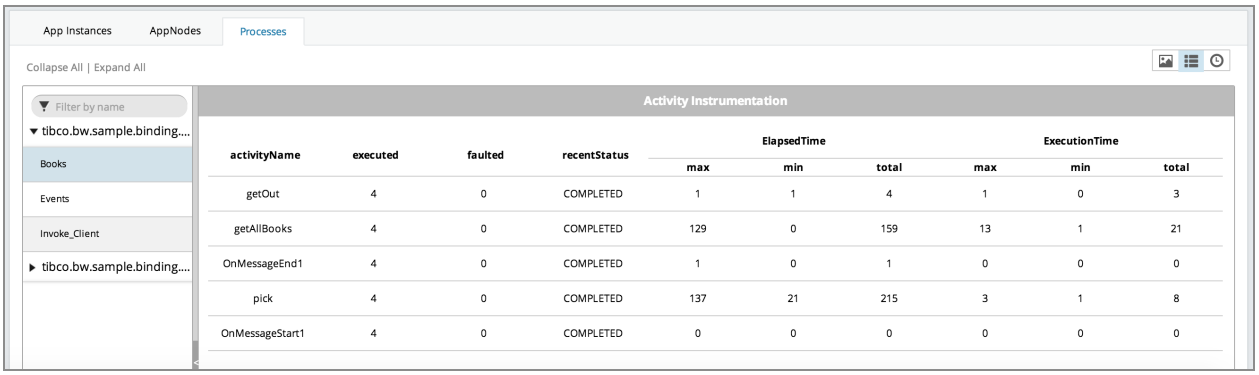


To view process instrumentation data, click an individual process. The Admin UI switches to the **Processes** tab, and the process diagram, along with process instrumentation data, displays.



View Activity Data

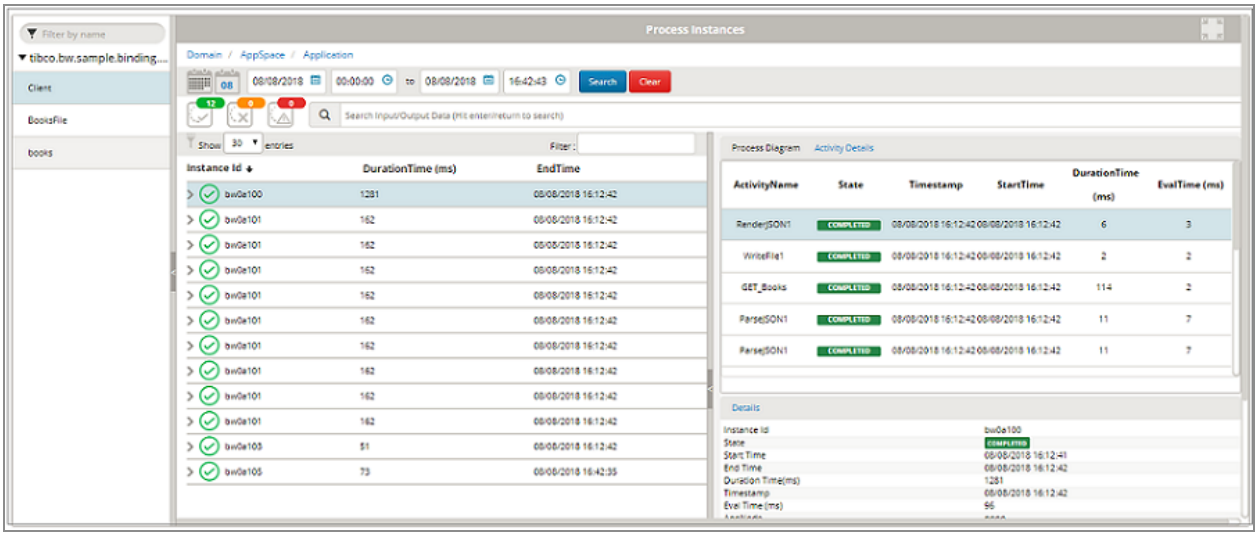
Select the **Application** tab, click the icon to switch to Monitoring View, and select the **Processes** tab to view the process diagram. From this point, you can view activity instrumentation data by clicking on an activity in the process diagram, or clicking the icon to the top-left corner of the **Processes** tab.



View Process and Activity Instance Data Logged to an External Database

Important: Before you can view process and activity instance data that has been logged to an external database, complete the steps outlined in the Writing Process Statistic Data to an External Database section, and enable process and activity instance statistics collection from the Admin UI.

Ensure you are on the **Application** tab, and click the icon to switch to Monitoring View. Expand an application to view the individual processes, and select one. Next, click the **Processes** tab, and click the icon, located at the top-left corner of the **Processes** tab, to view process instance details. Click the process instance ID to view its activity instances.



Process Execution Statistics

The following process execution statistics are collected by Logback.

Activity Instance Statistics

Activity Execution Statistics

Statistic	Description
Application Name	Name of application.
Application Version	Version of application.
Module Name	Name of BW module.
Module Version	Version of BW module.
Activity Name	Name of the activity.
Process Name	Name of the process.
Process Instance ID	Instance ID of the process.
Start Time	Activity start time (in milliseconds).
Duration Time	Total time (in milliseconds) taken by activity.
Eval Time	Total evaluation time (in milliseconds) taken by the activity.
Status	Status of activity, for example: Completed/Faulted/Canceled.
Domain	Name of the domain.
AppSpace	Name of the AppSpace.
AppNode	Name of the AppNode.

Process Instance Statistics

Process Instance Execution Statistics

Statistic	Description
Application Name	Name of application.
Application Version	Version of application.
Module Name	Name of BW module.
Module Version	Version of BW module.
Component Process Name	Name of process configured to a component. If the process is a non in-lined subprocess, this could be empty.
Job ID	Job ID of the process.
Parent Process Name	If the process is an in-lined subprocess, the name of the parent process.
Parent Process Instance ID	If the process is an in-lined subprocess, the instance ID of the parent process.
Process Name	Name of process.
Process Instance ID	Instance ID of the process.
Start Time	Process instance start time.
End Time	Process instance end time.
Duration Time	Total time (in milliseconds) taken by the process instance to finish.
Eval Time	Total evaluation time (in milliseconds) for all activities run for this process instance.

Statistic	Description
Status	Status of process instance, for example: Completed or Faulted

Note: Data is written as comma-separated values.

Integrating Execution Statistics Collection Using Logback

Edit the `logback.xml` to integrate execution statistics collection.

Procedure

1. Upload and deploy the application to an AppNode.
2. To enable activity execution statistics and process execution statistics, run the following commands from the Admin CLI:

   ```
   enablestats activityinstance appnameappversion
   ```

   ```
   enablestats processinstance appnameappversion
   ```

3. To disable activity execution statistics and process execution statistics, run the following commands from the Admin CLI:

   ```
   disablestats activityinstance appnameappversion
   ```

   ```
   disablestats processinstance appnameappversion
   ```

4. To retrieve execution statistics for a specific process, run the following command from the Admin CLI:

   ```
   enablestats -bp processname processinstance appnameappversion
   ```

5. To retrieve execution statistics for a specific activity in a specific process, run the

following command from the Admin CLI:

```
enablestats -bp processname -ba activityname activityinstance
appnameappversion
```

6. By default, statistics are collected in the following files:
   - Activity statistics: *AppNode_root*/stats/activitystats.csv
   - Process statistics: *AppNode_root*/stats/processstats.csv
7. To customize activity statistics collection, go to *AppNode_root*/logback.xml and configure the following appender:

```
<appender name="activityStatsFileAppender"
class="ch.qos.logback.core.rolling.RollingFileAppender">
   <File>../log/activitystats.log</File>
     <encoder>
       <Pattern>%msg%n</Pattern>
     </encoder>
   <rollingPolicy
class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
      <FileNamePattern>../log/activitystats.%d{yyyy-MM-
dd}.log</FileNamePattern>
   </rollingPolicy>
</appender>

<logger name="com.tibco.bw.statistics.activity" additivity="false">
  <level value="INFO"/>
  <appender-ref ref="activityStatsFileAppender" />
</logger>
```

   a. To write the log as a formatted HTML file, add the following file appender to the APPENDER: File Appender section of the logback.xml file.

```
<appender name="activityStatsFileAppender"
class="ch.qos.logback.core.FileAppender">
<File>../log/activitystats.html</File>
<encoder
class="ch.qos.logback.core.encoder.LayoutWrappingEncoder">
<layout
```

```
class="com.tibco.bw.logback.layout.ActivityExcecutionStatsHTML
Layout"/>
</encoder>
</appender>

<logger name="com.tibco.bw.statistics.activity"
additivity="false">
<level value="INFO"/>
<appender-ref ref="activityStatsFileAppender" />
</logger>
```

8. To customize process statistics collection, go to *AppNode_root*/logback.xml and configure the following appender:

```
<appender name="processinstanceStatsFileAppender"
class="ch.qos.logback.core.rolling.RollingFileAppender">
   <File>../log/processinstancestats.log</File>
     <encoder>
       <Pattern>%msg%n</Pattern>
     </encoder>
   <rollingPolicy
class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
      <FileNamePattern>../log/processinstancestats.%d{yyyy-MM-
dd}.log</FileNamePattern>
   </rollingPolicy>
</appender>

<logger name="com.tibco.bw.statistics.processinstance"
additivity="false">
  <level value="INFO"/>
  <appender-ref ref="processinstanceStatsFileAppender" />
</logger>
```

Results look similar to this:

```
bw0a104,paging.Process,Paging.application,1.0,2014-07-28 14:14:43:354,2014-07-28 14:14:53:357,10003,1,COMPLETED
bw0a105,paging.Process,Paging.application,1.0,2014-07-28 14:14:48:353,2014-07-28 14:14:58:357,10004,0,COMPLETED
bw0a106,paging.Process,Paging.application,1.0,2014-07-28 14:14:53:354,2014-07-28 14:15:03:355,10001,0,COMPLETED
bw0a107,paging.Process,Paging.application,1.0,2014-07-28 14:14:58:353,2014-07-28 14:15:08:356,10003,1,COMPLETED
bw0a108,paging.Process,Paging.application,1.0,2014-07-28 14:15:03:354,2014-07-28 14:15:13:356,10002,0,COMPLETED
bw0a109,paging.Process,Paging.application,1.0,2014-07-28 14:15:08:353,2014-07-28 14:15:18:354,10001,0,COMPLETED
bw0a10a,paging.Process,Paging.application,1.0,2014-07-28 14:15:13:353,2014-07-28 14:15:23:355,10002,1,COMPLETED
bw0a10b,paging.Process,Paging.application,1.0,2014-07-28 14:15:18:353,2014-07-28 14:15:28:354,10001,1,COMPLETED
bw0a10c,paging.Process,Paging.application,1.0,2014-07-28 14:15:23:354,2014-07-28 14:15:33:355,10001,0,COMPLETED
bw0a10d,paging.Process,Paging.application,1.0,2014-07-28 14:15:28:353,2014-07-28 14:15:38:356,10003,0,COMPLETED
bw0a10e,paging.Process,Paging.application,1.0,2014-07-28 14:15:33:354,2014-07-28 14:15:43:359,10005,1,COMPLETED
bw0a10f,paging.Process,Paging.application,1.0,2014-07-28 14:15:38:353,2014-07-28 14:15:48:354,10001,0,COMPLETED
bw0a10g,paging.Process,Paging.application,1.0,2014-07-28 14:15:43:352,2014-07-28 14:15:53:356,10004,0,COMPLETED
bw0a10h,paging.Process,Paging.application,1.0,2014-07-28 14:15:48:354,2014-07-28 14:15:58:355,10001,0,COMPLETED
bw0a10i,paging.Process,Paging.application,1.0,2014-07-28 14:15:53:354,2014-07-28 14:16:03:357,10003,0,COMPLETED
bw0a10j,paging.Process,Paging.application,1.0,2014-07-28 14:15:58:354,2014-07-28 14:16:08:355,10001,0,COMPLETED
bw0a10k,paging.Process,Paging.application,1.0,2014-07-28 14:16:03:354,2014-07-28 14:16:13:356,10002,0,COMPLETED
bw0a10l,paging.Process,Paging.application,1.0,2014-07-28 14:16:08:353,2014-07-28 14:16:18:355,10002,2,COMPLETED
bw0a10m,paging.Process,Paging.application,1.0,2014-07-28 14:16:13:353,2014-07-28 14:16:23:355,10002,0,COMPLETED
bw0a10n,paging.Process,Paging.application,1.0,2014-07-28 14:16:18:354,2014-07-28 14:16:28:358,10004,1,COMPLETED
bw0a10o,paging.Process,Paging.application,1.0,2014-07-28 14:16:23:353,2014-07-28 14:16:33:356,10003,0,COMPLETED
bw0a10p,paging.Process,Paging.application,1.0,2014-07-28 14:16:28:354,2014-07-28 14:16:38:355,10001,0,COMPLETED
bw0a10q,paging.Process,Paging.application,1.0,2014-07-28 14:16:33:353,2014-07-28 14:16:43:354,10001,0,COMPLETED
bw0a10r,paging.Process,Paging.application,1.0,2014-07-28 14:16:38:353,2014-07-28 14:16:48:354,10001,1,COMPLETED
bw0a10s,paging.Process,Paging.application,1.0,2014-07-28 14:16:43:353,2014-07-28 14:16:53:356,10003,0,COMPLETED
bw0a10t,paging.Process,Paging.application,1.0,2014-07-28 14:16:48:353,2014-07-28 14:16:58:354,10001,0,COMPLETED
bw0a10u,paging.Process,Paging.application,1.0,2014-07-28 14:16:53:353,2014-07-28 14:17:03:354,10001,0,COMPLETED
bw0a10v,paging.Process,Paging.application,1.0,2014-07-28 14:16:58:353,2014-07-28 14:17:08:354,10001,1,COMPLETED
bw0a1010,paging.Process,Paging.application,1.0,2014-07-28 14:17:03:353,2014-07-28 14:17:13:354,10001,0,COMPLETED
bw0a1011,paging.Process,Paging.application,1.0,2014-07-28 14:17:08:353,2014-07-28 14:17:18:357,10004,1,COMPLETED
bw0a1012,paging.Process,Paging.application,1.0,2014-07-28 14:17:13:353,2014-07-28 14:17:23:355,10002,1,COMPLETED
bw0a1013,paging.Process,Paging.application,1.0,2014-07-28 14:17:18:353,2014-07-28 14:17:28:355,10002,0,COMPLETED
bw0a1014,paging.Process,Paging.application,1.0,2014-07-28 14:17:23:352,2014-07-28 14:17:33:354,10002,0,COMPLETED
```

a. To write the log as a formatted HTML file, add the following file appender to the `APPENDER: File Appender` section of the `logback.xml` file.

```
<appender name="processinstanceStatsFileAppender"
class="ch.qos.logback.core.FileAppender">
   <File>../log/processinstancestats.html</File>
     <encoder
class="ch.qos.logback.core.encoder.LayoutWrappingEncoder">
       <layout
class="com.tibco.bw.logback.layout.ProcessInstanceStatsHTMLLay
out"/>
     </encoder>
</appender>

<logger name="com.tibco.bw.statistics.activity"
additivity="false">
<level value="INFO"/>
<appender-ref ref="processinstanceStatsFileAppender" />
</logger>
```

Results look similar to this:

Application Name	Application Version	Module Name	Module Version	Component Process Name	JobId	Parent Process Name	Parent ProcessInstanceId	Process Name	ProcessInstanceId	Start Time	End Time	Elapsed Time(Milliseconds)
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a101	-	-	paging.Process	bw0a101	2014-07-31 17:03:23:353	2014-07-31 17:03:33:354	10001
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a102	-	-	paging.Process	bw0a102	2014-07-31 17:03:28:353	2014-07-31 17:03:38:354	10001
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a103	-	-	paging.Process	bw0a103	2014-07-31 17:03:33:353	2014-07-31 17:03:43:355	10002
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a104	-	-	paging.Process	bw0a104	2014-07-31 17:03:38:353	2014-07-31 17:03:48:353	10000
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a105	-	-	paging.Process	bw0a105	2014-07-31 17:03:43:352	2014-07-31 17:03:53:354	10002
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a106	-	-	paging.Process	bw0a106	2014-07-31 17:03:48:353	2014-07-31 17:03:58:354	10001
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a107	-	-	paging.Process	bw0a107	2014-07-31 17:03:53:353	2014-07-31 17:04:03:353	10000
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a108	-	-	paging.Process	bw0a108	2014-07-31 17:03:58:353	2014-07-31 17:04:08:355	10002
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a109	-	-	paging.Process	bw0a109	2014-07-31 17:04:03:353	2014-07-31 17:04:13:353	10000
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a10a	-	-	paging.Process	bw0a10a	2014-07-31 17:04:08:353	2014-07-31 17:04:18:353	10000
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a10b	-	-	paging.Process	bw0a10b	2014-07-31 17:04:13:352	2014-07-31 17:04:23:354	10002
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a10c	-	-	paging.Process	bw0a10c	2014-07-31 17:04:18:352	2014-07-31 17:04:28:354	10002
Paging.application	1.0	Paging	1.0.0.qualifier	paging.Process	bw0a10d	-	-	paging.Process	bw0a10d	2014-07-31 17:04:23:353	2014-07-31 17:04:33:354	10001

9. Create a table in the database using following script or use an existing one.

```
create table PROCESS_INSTANCE_STAT_TABLE (APPLICATION_NAME varchar
(255),APPLICATION_VERSION varchar(255),MODULE_NAME varchar
(255),MODULE_VERSION varchar(255),COMPONENT_PROCESS_NAME varchar
(255),JOBID varchar(255),PARENT_PROCESS_NAME varchar(255),PARENT_
PROCESS_INSTANCEID varchar(255),PROCESS_NAME varchar(255),PROCESS_
INSTACEID varchar(255),START_TIME varchar(255),END_TIME varchar
(255),ELASPED_TIME varchar(255),EVAL_TIME varchar(255),STATUS
varchar(255));
```

Statistics is written either in the database or in the .csv file, but not both. If statistic is to be written in .csv, you need to put `statsprovider=false` in the `bwagent_xxx.json` file, and repush the configuration to the `bwaget.ini` file.

Enabling and Disabling Auditing Events

The auditing is used to build metrics and statistics for an application.

Configure the following BWEngine property in the `config.ini` file to enable and disable auditing at AppSpace and AppNode level.

- `bw.engine.disable.auditevent`: By default, the value of the property is false.

> **Note:** Restart the AppNode after configuring the property.

You can perform auditing without restarting an AppNode or an application with the help of the following CLI commands:

Command	Description
`enableauditevents`	Enable Audit Events for an AppNode.
`disableauditevents`	Disable Audit Events for an AppNode. **Note:** Once you disable audit events, metrics and statistics-related features of the TIBCO ActiveMatrix BusinessWorks™ Administrator do not work.

Enabling and Disabling Input and Output Data for Audit Events

Configure the following BWEngine property in the `config.ini` file to enable and disable input and output data for audit events at an AppSpace and an AppNode level:

- `bw.engine.enable.activity.input.output.data.for.audit.events`: By default, the value of the property is false.

> **Note:** Restart the AppNode after configuring these properties.

You can perform enable or disable input and output data for audit events without restarting an AppNode or an application by using the following CLI commands:

Command	Description
`enableinputoutputdataforauditevents`	Enable input output data for audit events for an AppNode.
`disableinputoutputdataforauditevents`	Disable input output data for audit events for an AppNode.

When using Statistics collection, the input and output data storage can also be disabled or enabled for audit events using the following BWEngine REST APIs:

- To disable:

  `http://<host>:<appnode port>/bwm/monitor.json/disableinputoutputdataforauditevents`

- To enable:

```
http://<host>:<appnode
port>/bwm/monitor.json/enableinputoutputdataforauditevents
```

Applying Security Policies

ActiveMatrix BusinessWorks includes a governance agent that enforces policies for ActiveMatrix BusinessWorks applications. Every installation of ActiveMatrix BusinessWorks includes the governance agent that facilitates the enforcement of cross-functional requirements such as security and compliance for your applications. The governance agent is disabled by default. In order to apply security policies, you must enable and configure the governance agent.

For information about enabling the governance agent with TEA, see Enabling the Governance Agent with TEA.

For information about enabling the governance agent using the AppSpace configuration file, see Enabling the Governance Agent Using the AppSpace Configuration File.

Note: Backwards compatibility is disabled by default for ActiveMatrix BusinessWorks 6.x applications using TIBCO ActiveMatrix Policy Director 2.0 to enforce security policies. To enable backwards compatibility, add `bw.governance.pd.compatibility.mode=true` to the existing AppSpace configuration file `appspace_config.ini` (in the root of the AppSpace folder), or the AppSpace configuration template file, `appspace_config.ini_template`, in `<BW_HOME>\config\`.

Enabling the Governance Agent Using the Admin UI

The governance agent is disabled by default. In order to apply security policies, enable the `bw.governance.enabled` property.

Before you begin

Complete the following tasks:

- The BWAdmin mode must be set to enterprise.
- The TIBCO Enterprise Administrator server must be running.

- The BWAgent TEA agent must be registered with the TEA server.

Follow these steps to enable the governance agent using the Admin UI:

Procedure

1. Open a web browser and go to the TEA URL. Sign in, by typing **admin** for the username and **admin** for the password.

   BusinessWorks is displayed in the Products list.

2. Click the BusinessWorks icon to go to ActiveMatrix BusinessWorks.

   The Domain Management page displays.

3. Click the **AppSpace** icon

to open the AppSpace page.

4. Click the AppSpace hosting your application.

5. On the AppSpaces page, click the **Configure** icon

to view a list of AppSpace properties you can modify. You can also click **Configure** icon in the upper right of the AppSpace page.

6. Scroll down to find the `bw.governance.enabled` property. By default, the value is **false**.

7. Type **true** and click the **Check** icon to enable the governance agent. Ensure you enter the value under the **Current Value** column.

   > **Note:** Ensure the property `bw.governance.jms.server.url` does not have a value. The property is used to specify the JMS server URL used to communicate with the TIBCO Policy Director Administrator.

8. Stop and restart the AppSpace to apply the changes.

Result

The governance agent is enabled.

Enabling the Governance Agent Using an AppSpace Configuration File

The governance agent within each AppNode is disabled by default. Enable it by setting properties within their respective `config.ini` files.

Enabling the Governance Agents in the AppNodes of an AppSpace

Each AppNode in ActiveMatrix BusinessWorks includes a governance agent that enforces policies for ActiveMatrix BusinessWorks applications. The governance agents are disabled by default. In order to apply security policies, you must enable these governance agents and configure the environment as described below.

To enable governance on an AppSpace, configure the governance agent property on the AppSpace by following these steps:

1. Copy the existing AppSpace configuration file `appspace_config.ini` that is in the root of the AppSpace folder, or the AppSpace configuration template file, `appspace_config.ini_template` that is in `<BW_HOME>\config\` to a temporary location.

   Note: Do not modify the original AppSpace configuration file, `config.ini` in the root of the AppSpace folder, or the AppSpace configuration template file, `appspace_config.ini_template` file. Instead, update the copy of the file that is in the temporary location.

2. Edit the configuration file in the temporary location to set the following properties.

   Note:
   - Set the value for `bw.governance.enabled` to true to enable the governance agent. If no ActiveMatrix BusinessWorks applications are using TIBCO ActiveMatrix Policy Director to enforce security policies, comment out the property `bw.governance.jms.server.url`.
   - If the TIBCO ActiveMatrix Policy Director is already set up, ensure that the JMS server properties specified in the AppSpace configuration file match the JMS server configured in the TIBCO ActiveMatrix Policy Director server.

```
#
------------------------------------------------------------------
------
# Section: BW Governance Agent & SPM Configuration. The properties
in
# this section are applicable to Governance Agent and the
Governance SPM
# EventSubscriber that is run within a BW AppNode.
#
------------------------------------------------------------------
------
# Enable or disable the governance agent. This property is optional
and
# it specifies whether the governance agent should be enabled or
disabled
# in the AppNode. The supported values are: true or false. The
default
# value is “false”.
bw.governance.enabled=true

# BW Governance Agent JMS URL. This property is optional and it is
used
# to specify the JMS server URL used to communicate with the
# TIBCO Policy Director Administrator. If this property is not set,
then
# the BW Governance agent will not attempt to connect to the JMS
server.
# The URL is expected to start with 'tcp://' or 'ssl://' and the
failover
# URLs can be specified as a ',' or '+' separated list.
bw.governance.jms.server.url=tcp://localhost:7222

# BW Governance Agent JMS User Name. This property is required if
the
# Governance Agent JMS URL is specified.
bw.governance.jms.server.username=admin

# BW Governance Agent JMS User Password. This property is required
if the
# Governance Agent JMS URL is specified.
bw.governance.jms.server.password=

# BW Governance Agent JMS SSL connection trust store type. This
```

```
property
# is required if the JMS server protocol is ssl. The supported
values are
# 'JKS'and 'JCEKS'. The default value is 'JKS'
bw.governance.jms.ssl.trust.store.type=JKS

# BW Governance Agent JMS SSL connection trust store location. This
# property is required if the JMS server protocol is ssl.
bw.governance.jms.ssl.trust.store.location=

# BW Governance Agent JMS SSL connection trust store password. This
# property is required if the JMS server protocol is ssl. The
password
# may be clear text or supplied as an obfuscated string.
bw.governance.jms.ssl.trust.store.password=

# BW Governance Agent JMS Connection attempt count. This property
is
# required if the Governance Agent JMS URL is specified and it
specifies
# the number of JMS connection attempts the Governance Agent will
make.
# The default value is '120'.
bw.governance.jms.reconnect.attempt.count=120

# BW Governance Agent JMS Connection attempt timeout. This property
is
# required if the Governance Agent JMS URL is specified and it
specifies
# the timeout between the attempt to reestablish connection to the
JMS
# server. The default value is '500'.
bw.governance.jms.reconnect.attempt.timeout=500

# BW Governance Agent JMS Connection attempt delay. This property
is
# required if the Governance Agent JMS URL is specified and it
specifies
# the delay in milliseconds between attempts to establish
reestablish
# connection to the JMS server. The default value is '500'.
bw.governance.jms.reconnect.attempt.delay=500
```

```
# BW Governance Agent JMS receiver queue name. This property is
required
# if the Governance Agent JMS URL is specified and it specifies
receiver
# queue name for the governance agent and administrator
communication.
# The default value is 'queue.bw.governance.agent.bw.default’.

bw.governance.jms.queue.receiver.name=queue.governance.agent.bw.def
ault

# BW Governance Agent JMS sender queue name. This property is
required
# if the Governance Agent JMS URL is specified and it specifies the
# sender queue name for the governance agent and administrator
# communication. It must match the value specified in the Policy
Director
# Administrator configuration.
# The default value is 'governance.de.bw.default’.
bw.governance.jms.queue.sender.name=governance.de.bw.default

# BW Governance Agent JMS JNDI custom property. This property is
optional
# and it provides the ability to specify custom property for the
# JMS JNDI Initial Context. For example to provide a custom
property
# called "myProperty" for the JNDI Initial Context, then specify
# a property "bw.governance.jms.application.property.myProperty=".

#bw.governance.jms.application.property.<UserCustomProperty>=<userV
alue>
# BW Governance Agent Shared Resource lookup. This property is
optional
# and it provides ability for the Governance Agent to lookup shared

# resources.
# bw.governance.sr.WSSConfiguration=com.tibco.trinity.runtime.core.
# provider.authn.wss
```

3. Restart the AppSpace from the TIBCO ActiveMatrix BusinessWorks agent user interface in TEA.

OpenTelemetry

OpenTelemetry is an open source, vendor neutral standard for distributed systems that can be used to track the current state of the job. OpenTelemetry is a set of APIs, SDKs, toolings, and integrations designed to create and manage telemetry data such as traces and metrics.

Note: OpenTelemetry does not support checkpointing. If a system fails after the checkpoint activity, then traces of the activities before the **Checkpoint** activity are not seen after restarting the system.

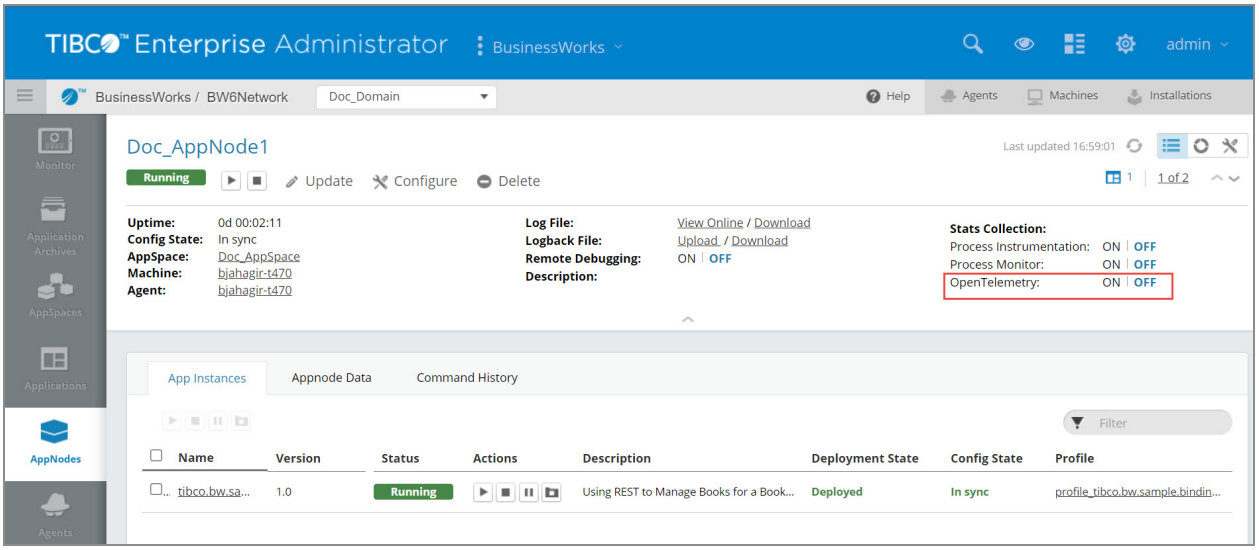
For more information about OpenTelemetry, see the OpenTelemetry documentation.

Enabling or Disabling OpenTelemetry

OpenTelemetry can be enabled or disabled through the Admin UI or the `BW_JAVA_OPTS` environment variable.

Admin UI

You can enable or disable OpenTelemetry at an AppNode Page 2 level.



`BW_JAVA_OPTS` Environment Variable

When running the application to enable or disable OpenTelemetry at an AppNode level, in the `BW_JAVA_OPTS` environment variable, configure the following BWEngine property :

`bw.engine.opentelemetry.enable=true`

> **Note:** By default, the property is false.

Dynamically Enabling and Disabling OpenTelemetry

You can enable and disable OpenTelemetry without restarting an AppNode or an application with the help of the following Admin CLI commands:

Command	Description
`enableopentelemetry`	Enable OpenTelemetry for an AppNode.
`disableopentelemetry`	Disable OpenTelemetry for an AppNode.

The following table describes how you can enable trace, metric, or both the variants simultaneously by setting up the BWEngine properties accordingly:

Properties	Trace	Metric
`bw.engine.opentelemetry.enable=true` `bw.engine.opentelemetry.trace.enable=true` or blank `bw.engine.opentelemetry.metric.enable=false` or blank	Enable	Disable

`bw.engine.opentelemetry.enable=true`
`bw.engine.opentelemetry.trace.enable=false`
`bw.engine.opentelemetry.metric.enable=true`
	Disable	Enable

`bw.engine.opentelemetry.enable=true`
`bw.engine.opentelemetry.trace.enable=true` or blank
`bw.engine.opentelemetry.metric.enable=true`
	Enable	Enable

Configuring OpenTelemetry with OpenTelemetry-Collector

1. Set up the OpenTelemetry-collector service. You can further integrate OpenTelemetry with a tracing service provider that is compliant with OpenTelemetry.
2. To configure OpenTelemetry with OpenTelemetry-collector, in the AppNode's `config.ini` file, set the `bw.engine.opentelemetry.enable=true` property to true.

> **Note:** The OpenTelemetry via OpenTelemetry-Collector is the recommended approach.

To configure OpenTelemetry native properties, set the `bw.opentelemetry.autoConfigured` system property to `true`. Thereafter, if this property is set to true, you can use the environment variables listed at https://opentelemetry.io/docs/specs/otel/configuration/sdk-environment-variables/.

To send data over the HTTP protocol for OpenTelemetry traces and metrics, set the `bw.engine.opentelemetry.http.protocol` system property to `true`.

To configure AWS_XRAY for OpenTelemetry traces and metrics, set the `bw.engine.opentelemetry.enable`, `bw.opentelemetry.aws.xrayIdGenerator`, and `bw.engine.opentelemetry.metric.enable` system property to `true`.

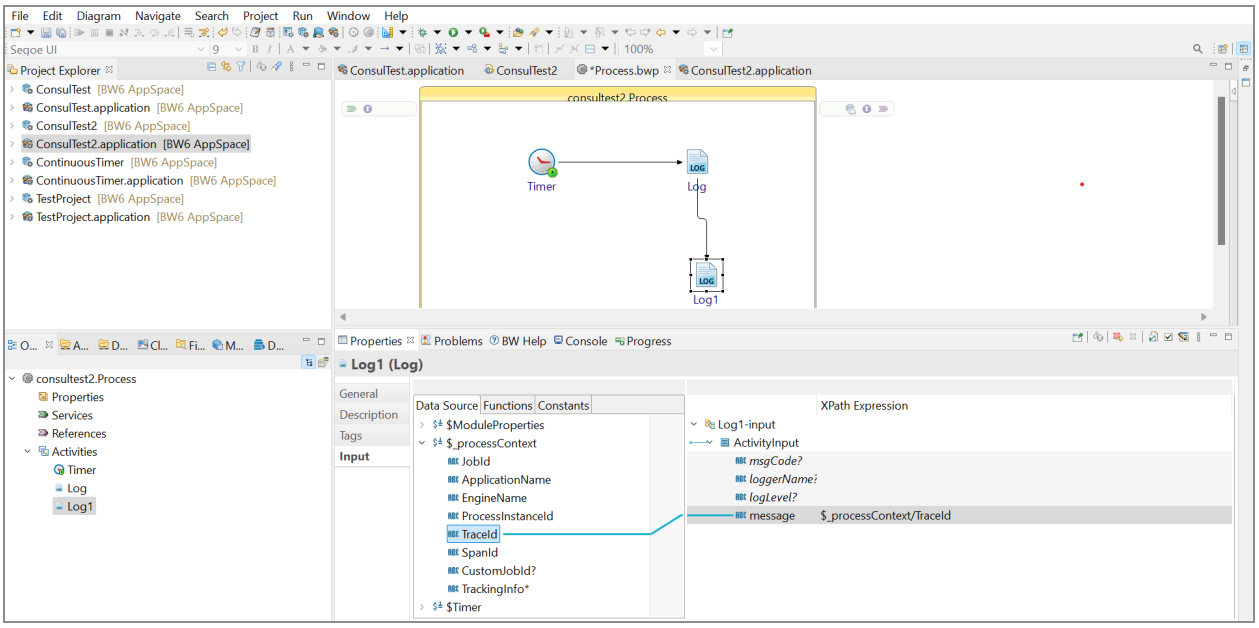
To enable logging for the OpenTelemetry traces and metrics, set the `io.opentelemetry` logger in the `logback.xml` file. On enabling this logger, all the detailed information about traces and metrics is available in the logs.

Example:

```
<logger name="io.opentelemetry">
      <level value="ALL"/>
</logger>
```

> **Note:** The OpenTelemetry's TraceId and SpanId are available in BW logs (TIBCO Business Studio for BusinessWorks and Runtime). To get the TraceId and SpanId in the log, you must enable OpenTelemetry.

The OpenTelemetry TraceId and SpanId parameters can be mapped from an application's ProcessContext to any output activity such as Log or Write File.



Traces

ActiveMatrix BusinessWorks supports all OpenTelemetry-compliant telemetry backends to display a span for each activity and process instance during job execution. Span corresponds to a process instance as well as an activity instance that has information such as ActivityName, JobID, and process instance ID. For every process instance, a root span is created and all the activity instances are child spans of it.

Traces represent multiple related process instance spans.

Note: In case of HTTP palette, JMS palette, REST binding, and SOAP binding, client, and server process instances are shown in one trace, whereas for all other palettes, every process instance is a trace.

Note: Traces can be enabled by enabling the `bw.engine.opentelemetry.enable` property. By default, it is false.

You can configure the following properties specific to OpenTelemetry:

Property	Value	Description
bw.engine.opentelemetry.span.processor	Possible values are SPAN or BATCH. The default value is BATCH.	Configure Span Processor type.
bw.engine.opentelemetry.span.processor.delay	Value in milliseconds	Sets the delay interval between two consecutive exports.
bw.engine.opentelemetry.span.processor.timeout	Value in milliseconds	Sets the maximum time an export is allowed to run before being canceled.
bw.engine.opentelemetry.span.processor.batch.size	Integer value in kb.	Sets the maximum batch size for every export. This must be smaller or equal to maxQueuedSpans.
bw.engine.opentelemetry.span.processor.queue.size	Queue size in kb	Sets the maximum number of spans that are kept in the queue before start dropping. More memory than this value may be allocated to optimize queue access.
bw.engine.opentelemetry.span.sampler	ON, OFF, 0.0 to 1.0. The default value is ON.	Configure Span Sampler type.
bw.engine.opentelemetry.span.exporter	OTLP-GRPC	This property helps you to set a custom exporter injected as a service. The value of this property should be the component name of the service.

Property	Value	Description
		For the Jaeger exporter, the value for this property should be set to `com.tibco.bw.opentelemetry.exporter.jaeger`.
`bw.engine.opentelemetry.span.exporter.endpoint`	http://<host>:<port>	Sets the OTLP or Jaeger endpoint to connect to. **Note:** In case of TIBCO BusinessWorks Container Edition, it is mandatory to set this property.
`bw.engine.opentelemetry.span.exporter.timeout`	Value in milliseconds	Sets the maximum time to wait for the collector to process an exported batch of spans.

Supported tags for querying on OpenTelemetry

Currently, the following tags are supported for querying on OpenTelemetry:

Tag	Description
SpanInitiator	Name of the process starter activity.
DeploymentUnitName	Name of the application.
DeploymentUnitVersion	Version of the application.
AppnodeName	Name of an AppNode on which an application is running.
Hostname	Name of the machine on which a TIBCO ActiveMatrix BusinessWorks™ application is running. This tag is applicable for Jaeger exporter UI.

Tag	Description
IP	IP address. This tag is applicable for Jaeger exporter UI.
ActivityName	Name of an activity in a process.
ActivityID	ID of an activity.
ProcessInstanceId	Process instance ID.
JobId	Job ID of the process.
ProcessName	Name of the process displayed for starter activities.

OpenTelemetry via Jaeger Span Exporter

1. Set up a Jaeger service.
2. To configure OpenTelemetry with Jaeger span exporter by using the following properties:

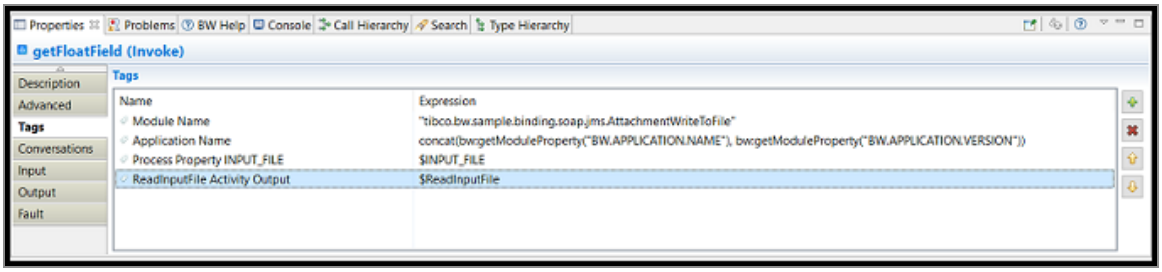
```
bw.engine.opentelemetry.enable=true

bw.engine.opentelemetry.span.exporter=com.tibco.bw.opentelemetry.exporter.jaeger

bw.engine.opentelemetry.span.exporter.endpoint=http://localhost:14250
```

Caution: With OpenTelemetry Span Exporter, the tags under process detail such as hostname, IP, Jaeger version are not displayed on the Jaeger UI. If you use the Jaeger exporter service instead of the default OpenTelemetry exporter service, the tags are visible on the Jaeger UI.

By default, the OpenTelemetry traces by using Jaeger Span Exporter and OpenTelemetry Collector are available on Jaeger UI at `http://localhost:16686/`.

Custom Tags for OpenTelemetry

For OpenTelemetry, you can add custom tags. To add custom tags, use the **Tags** tab added in each activity in TIBCO Business Studio for BusinessWorks.



You can add **Expression** such as hard-coded values and XPath expressions for custom tags.

At run time, an asterisk (*) prefix is added for the names of the custom tags. It avoids overriding of the predefined engine tags.

OpenTelemetry Tags from Palettes

To get more information about the current job in execution, activity level tags are also supported. These tags are predefined tags.

The following sections show the list of predefined tags supported by each activity:

Basic Activities Palette

Activity name	Supported Tags
Invoke	• Service name • Operation Name

General Palette

Activity Name	Supported Tags
Confirm	Confirm Event
Call Process	• Spawned • Called Process Name
External Command	• Command • Environment
Log	Log Level
Sleep	Interval In MilliSec

File Palette

Activity Name	Supported Tags
Copy File	• From File • To File
Create File	File Name
File Poller	• File Name • Polling Interval(sec)
List Files	• File Name Pattern • Number Of Files • Mode
Read File	• File Name • Content Style

Activity Name	Supported Tags
Remove File	File Name
Rename File	• From File • To File
Write File	• File Name • Write As
Wait For File Change	• File Name • Polling Interval(sec)

FTP Palette

Activity Name	Supported Tags
FTP Change Default Directory	• peer.hostname • peer.port
FTP Delete File	• peer.hostname • peer.port
FTP Dir	• peer.hostname • peer.port
FTP Get	• peer.hostname • peer.port
FTP Get Default Directory	• peer.hostname • peer.port
FTP Make Remote Directory	• peer.hostname • peer.port

Activity Name	Supported Tags
FTP Put	• peer.hostname • peer.port
FTP Quote	• peer.hostname • peer.port
FTP Remove Remote Directory	• peer.hostname • peer.port
FTP Rename File	• peer.hostname • peer.port
FTP SYS Type	• peer.hostname • peer.port

HTTP Palette

Activity Name	Supported Tags
HTTP Receiver	• net.peer.name • net.peer.port • http.url • span.kind • error • ErrorMessage
Send HTTP Request	• span.kind • http.url • HTTPRequestQuery • HTTPPostDataType

Activity Name	Supported Tags
	• HTTPCookiePolicy • http.method • IsSecureHTTP • error • ErrorMessage • ErrorCode • ErrorStatus
Send HTTP Response	• span.kind • http.status_code • net.peer.name • net.peer.port • http.method • peer.ipv4 • HttpServerProtocol • ContentType • IsSecureHTTP • error • HTTPServerErrorMessage • HTTPServerErrorCode • ErrorCode • ErrorMessage
Wait For HTTP Request	• net.peer.name • net.peer.port • http.url • span.kind

Activity Name	Supported Tags
	• error • ErrorMessage

Java Palette

Activity Name	Supported Tags
Java Invoke	• Class Name • Method Name • CleanUp method • Global Instance • Method Return • IsMultipleOutput • Construct Declared • Cache Declared
Java To XML	• Class Name • Constructor Declared • Cache Declared
XML To Java	Class Name

JDBC Palette

Activity Name	Supported Tags
JDBC Call Procedure	• ActivitySharedResourceURL • ActivityIsOverrideSharedResource • ActivityOverrideSharedResourceUR

Activity Name	Supported Tags
	• ActivityInTransaction • ActivityExecutionStatus
JDBC Query	• ActivitySharedResourceURL • ActivityIsOverrideSharedResource • ActivityOverrideSharedResourceUR • ActivityInTransaction • ActivityExecutionStatus
JDBC Update	• ActivitySharedResourceURL • ActivityIsOverrideSharedResource • ActivityOverrideSharedResourceURL • ActivityInTransaction • ActivityExecutionStatus
SQL Direct	• ActivitySharedResourceURL • ActivityIsOverrideSharedResource • ActivityOverrideSharedResourceURL • ActivityInTransaction • ActivityExecutionStatus

JMS Palette

Activity Name	Supported Tags
Get JMS Queue Message	• messaging.destination • MessagingStyle • MessageType • AcknowledgementMode

Activity Name	Supported Tags
JMS Receive Message	• messaging.destination • MessagingStyle • MessageType • span.kind
JMS Request Reply	• messaging.destination • MessagingStyle • MessageType • span.kind
JMS Send Message	• messaging.destination • MessagingStyle • MessageType • span.kind
Reply to JMS Message	• MessagingStyle • MessageType • span.kind • ReplyQueue
Wait for JMS Request	• messaging.destination • MessagingStyle • MessageType

Mail Palette

Activity Name	Supported Tags
Receive mail	• peer.hostname • peer.port

Activity Name	Supported Tags
	• From Address • Reply To Address • To Address
Send Mail	• peer.hostname • peer.port • From Address • Reply To Address • To Address • CC Address • BCC Address • Sent Date

Parse Palette

Activity Name	Supported Tags
Mime Parser	• InputStyle • OutputStyle
Parse Data	• FormatType • Encoding • LineLength • SkipBlankLines • ColumnSeperator • StringValue or FileName - Depending on input type • NumberOfRecord

Activity Name	Supported Tags
Render Data	• FormatType • LineLength • ColumnSeperator • FillCharacter

REST and JSON Palette

Activity Name	Supported Tags
Invoke REST API	• http.status_code • http.url • net.peer.name • net.peer.port • http.method • error • ErrorType • ErrorMessage
Parse JSON	• SchemaType • OutputRootElementName • IsBadgerfishEnabled • error • ErrorType • ErrorMessage
Render JSON	• IsJsonRenderException - This tag is populated only when some exception occurs. • SchemaType • RemoveRoot

Activity Name	Supported Tags
	• IsBadgerfishEnabled • error • ErrorType • ErrorMessage
Transform JSON	• error • ErrorType • ErrorMessage

TCP Palette

Activity Name	Supported Tags
Read TCP Data	• Data Type • Timeout • net.peer.name • net.peer.port
TCP Open Connection	• net.peer.name • net.peer.port
Wait For TCP Request	• net.peer.name • net.peer.port
Write TCP Data	• Data Type • net.peer.name • net.peer.port

XML Palette

Activity Name	Supported Tags
Parse XML	• IsOutputValidationEnabled • Input Style • error • ErrorType • ErrorMessage
Render XML	• IsInputValidationEnabled • Encoding • OutputStyle • DefaultNamespaceFormat • error • ErrorType • ErrorMessage
Transform XML	• InputOutputStyle • StyleSheet • error • ErrorType • ErrorMessage

OpenTelemetry Tags from SOAP Bindings

The following tags are supported for SOAP service and reference binding. Here, **Invoke** activity represents client-side tags and **Receive** activity represents server-side tags.

SOAP with HTTP

Side	Supported Tags
Service	• RequestURI • TransportType • http.method • peer.hostname • peer.port
Client	• TransportType • LocationURI • AttachementStyle • WSDLPort • ServiceName • OperationName

SOAP with JMS

Side	Supported Tags
Service	• ReplyTo • span.kind • messaging.destination • MessagingStyle • MessageType • Operation
Client	• TransportType • EndpointReference • ReplyTo • MessagingStyle

Side	Supported Tags
	• Service Name • Operation Name • messaging.destination • span.kind • MessageType

OpenTelemetry Tags from REST Binding

The following tags are supported for REST service and reference binding. Here, **Invoke** activity represents client-side tags and **Receive** activity represents server-side tags.

Side	Supported tags
Service	• http.url • isUsingSSL • error • errorMessage • errorStatus • net.peer.port • span.kind • net.peer.name • clientResponseFormat • http.method
Client	• http.url • isUsingSSL • error • errorMessage • errorStatus

Side	Supported tags
	• net.peer.port • http.status_code • span.kind • net.peer.name • isRequestBuffered • contentType • http.method

Metrics

ActiveMatrix BusinessWorks can export metrics data to OpenTelemetry that can be leveraged by the OpenTelemetry backend-supported client.

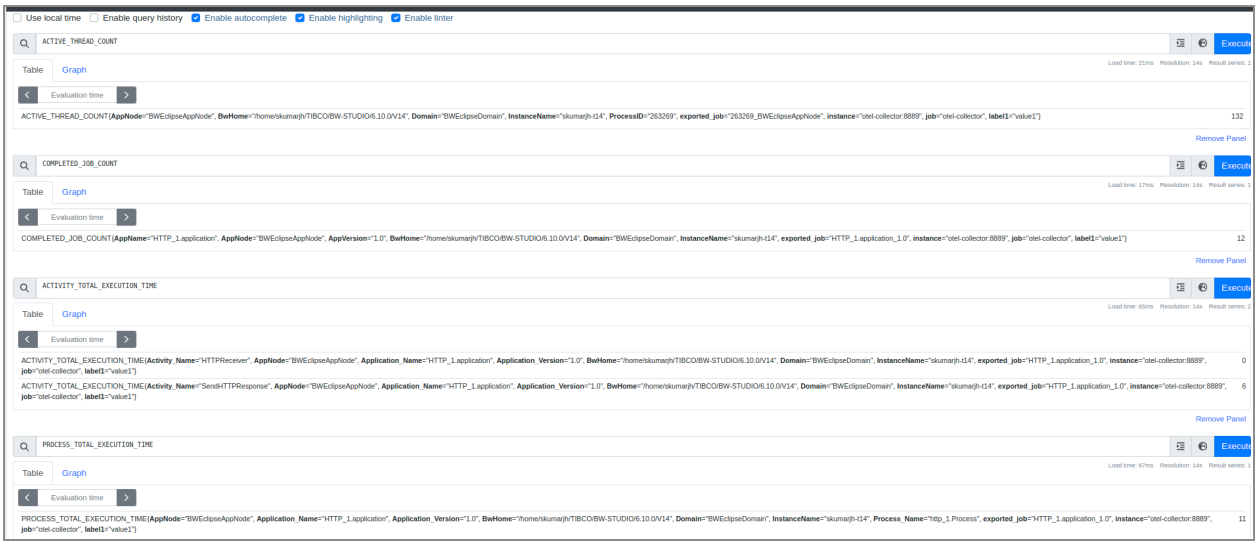
The following metrics data are sent to OpenTelemetry:

- App data (TOTAL_JOB_COUNT, etc.)
- System data (ACTIVE_THREAD_COUNT, etc.)
- Process and Activity data (ACTIVITY_MAX_ELAPSED_TIME, etc.)

The following properties must be enabled for exporting the metrics data to OpenTelemetry:

- `bw.engine.opentelemetry.enable=true`
- `bw.engine.opentelemetry.metric.enable=true`

Note:
- Metrics (fields and their values) are displayed only if the relevant data is available.
- To enable the process and activity data in the OpenTelemetry metric, first enable the Process Instrumentation data property.



The `bw.engine.opentelemetry.metric.exporter.endpoint` property is used to set up the OpenTelemetry for metrics with remote machines or custom endpoints. When the OpenTelemetry Collector and the ActiveMatrix BusinessWorks application are running on two different machines, this property is added to get the metrics exported to the OpenTelemetry Collector at a specified IP/Host and Port.

If this property is not provided, the OpenTelemetry Collector picks `localhost:4317` as a default endpoint.

For example, `bw.engine.opentelemetry.metric.exporter.endpoint=http://<host ip>:<port>`.

List of Ports

This is a list of ports that are used.

List of Ports

Port Description	Default Value	Configuration
External database port. Applies to BWAgent technology type of Database/EMS.	5432	`bw.agent.technology.dbems.db.connectionURL` property in `BW_HOME\config\bwagent.ini`
The internal HTTP communication port that the Thor engine uses to communicate with the BWAgent to send the status of AppNodes and applications. Update this property to specify a port to start the internal server on.	56565	`bw.appnode.agent.http.communication.port` property in `BW_HOME\config\bwagent.ini`
AppNode HTTP management port. Must be unique across all defined AppNodes on the same machine.	User-specified	`-httpPort` option in BWAdmin `create` command for AppNode creation or **HTTP Port** option in Create AppNode dialog in Admin UI.
AppNode remote debug port	User-specified	`-port` option in the BWAdmin `enabledebugport` command for remote debugging of an AppNode.
OSGi console port. Must be unique.	User-specified	`port` argument in the BWAdmin `enableconsole` command for enabling OSGi port for AppNode. Can also be specified with `-osgiPort` option when

Port Description	Default Value	Configuration
		AppNode created with `create` command.
• EMS server port for group configuration. • EMS server port for Database/EMS BWAgent technology type.	7222	Engine: `bw.engine.groupProvider.qin.EMSServerUrl` property in `BW_HOME\config\appspace_config-ini_template` Technology type: `bw.agent.technology.dbems.db.ems.serverURL` property in `BW_HOME\config\bwagent.ini`
Web server HTTP port for Swagger UI.	5555	Not configurable
Web server HTTP port	8079	`bw.agent.http.port` property in *`BW_HOME`*`\config\bwagent.ini`
Web server HTTPS port	8886	`bw.agent.https.port` property in `BW_HOME\config\bwagent.ini`
BWAgent TEA agent port	9091	`bw.agent.tea.agent.port` property in `BW_HOME\config\bwagent.ini`
BWAgent TEA agent shutdown port	5678	`bw.agent.tea.agent.shutdown.port` property in *`BW_HOME`*`\config\bwagent.ini`
TEA listen port	8777	`tea.http.port` property in *`TEA_CONFIG_HOME`*`\tibco\cfgmgmt\conf\tea.conf`
TEA SSH port	2222	`tea.shell.port` property in *`TEA_CONFIG_HOME`*`\tibco\cfgmgmt\conf\tea.conf`

TIBCO Documentation and Support Services

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

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the Product Documentation website, mainly in HTML and PDF formats.

The Product Documentation website is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The following documentation for this product is available on the TIBCO ActiveMatrix BusinessWorks™ page:

- *TIBCO ActiveMatrix BusinessWorks™ Release Notes*
- *TIBCO ActiveMatrix BusinessWorks™ Installation*
- *TIBCO ActiveMatrix BusinessWorks™ Application Development*
- *TIBCO ActiveMatrix BusinessWorks™ Bindings and Palettes Reference*
- *TIBCO ActiveMatrix BusinessWorks™ Concepts*
- *TIBCO ActiveMatrix BusinessWorks™ Error Codes*
- *TIBCO ActiveMatrix BusinessWorks™ Getting Started*
- *TIBCO ActiveMatrix BusinessWorks™ Maven Plug-in*
- *TIBCO ActiveMatrix BusinessWorks™ Migration*
- *TIBCO ActiveMatrix BusinessWorks™ Performance Benchmarking and Tuning*
- *TIBCO ActiveMatrix BusinessWorks™ REST Implementation*
- *TIBCO ActiveMatrix BusinessWorks™ Refactoring Best Practices*
- *TIBCO ActiveMatrix BusinessWorks™ Samples*

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.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the TIBCO Ideas Portal. For a free registration, go to TIBCO Community.

Legal and Third-Party Notices

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

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

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

TIBCO, the TIBCO logo, the TIBCO O logo, ActiveMatrix BusinessWorks, ActiveSpaces, Business Studio, TIBCO Business Studio, TIBCO Designer, TIBCO Enterprise Administrator, Enterprise Message Service, Rendezvous, and TIBCO Runtime Agent 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 © 2001-2024. Cloud Software Group, Inc. All Rights Reserved.