Installing ThingWorx 7.2

Version 1.5

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

User and training guides and related documentation from PTC Inc. and its subsidiary companies (collectively “PTC”) are subject to the copyright laws of the United States and other countries and are provided under a license agreement that restricts copying, disclosure, and use of such documentation. PTC hereby grants to the licensed software user the right to make copies in printed form of this documentation if provided on software media, but only for internal/personal use and in accordance with the license agreement under which the applicable software is licensed. Any copy made shall include the PTC copyright notice and any other proprietary notice provided by PTC. Training materials may not be copied without the express written consent of PTC. This documentation may not be disclosed, transferred, modified, or reduced to any form, including electronic media, or transmitted or made publicly available by any means without the prior written consent of PTC and no authorization is granted to make copies for such purposes.

Information described herein is furnished for general information only, is subject to change without notice, and should not be construed as a warranty or commitment by PTC. PTC assumes no responsibility or liability for any errors or inaccuracies that may appear in this document.

The software described in this document is provided under written license agreement, contains valuable trade secrets and proprietary information, and is protected by the copyright laws of the United States and other countries. It may not be copied or distributed in any form or medium, disclosed to third parties, or used in any manner not provided for in the software licenses agreement except with written prior approval from PTC.

UNAUTHORIZED USE OF SOFTWARE OR ITS DOCUMENTATION CAN RESULT IN CIVIL DAMAGES AND CRIMINAL PROSECUTION. PTC regards software piracy as the crime it is, and we view offenders accordingly. We do not tolerate the piracy of PTC software products, and we pursue (both civilly and criminally) those who do so using all legal means available, including public and private surveillance resources. As part of these efforts, PTC uses data monitoring and scouring technologies to obtain and transmit data on users of illegal copies of our software. This data collection is not performed on users of legally licensed software from PTC and its authorized distributors. If you are using an illegal copy of our software and do not consent to the collection and transmission of such data (including to the United States), cease using the illegal version, and contact PTC to obtain a legally licensed copy.

Important Copyright, Trademark, Patent, and Licensing Information: See the About Box, or copyright notice, of your PTC software.

United States Governments Rights
PTC software products and software documentation are “commercial items” as that term is defined at 48 C.F.R. 2.101. Pursuant to Federal Acquisition Regulation (FAR) 12.212 (a)-(b) (Computer Software) (MAY 2014) for civilian agencies or the Defense Federal Acquisition Regulation Supplement (DFARS) at 227.7202-1 (a) (Policy) and 227.7202-3 (a) (Rights in commercial computer software or commercial computer software documentation) (FEB 2014) for the Department of Defense, PTC software products and software documentation are provided to the U.S. Government under the PTC commercial license agreement. Use, duplication or disclosure by the U.S. Government is subject solely to the terms and conditions set forth in the applicable PTC software license agreement.

PTC Inc., 140 Kendrick Street, Needham, MA 02494 USA

Document Revision History

Revision Date	Version	Description of Change
January 03, 2017	1.5	Updated optional password encryption info.
September 07, 2016	1.4	Updated “production environments” wording
September 1, 2016	1.3	Added Encrypting the PostgreSQL Password sections
August 24, 2016	1.2	Updated RHEL command
August 02, 2016	1.1	Updated RHEL commands
July 28, 2016	1.0	Initial version for 7.2

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Installing ThingWorx

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Installing ThingWorx Core

ThingWorx Core is currently supported on Windows, Ubuntu, and Red Hat Enterprise Linux.

Prerequisites

Prerequisite software includes Apache Tomcat and Oracle Java. PostgreSQL is also required if you are not using H2. If you are installing ThingWorx for the first time, this document provides step-by-step installation instructions for your environment.

If you are upgrading to a newer version, refer to the Upgrading ThingWorx guide.

Database Options: PostgreSQL or H2

With ThingWorx 7.2, you can use PostgreSQL (with an optional High Availability layer) or H2 for your data solution. If you are upgrading to 7.2, the following download package options are available when obtaining the thingworx.war from PTC Software Downloads:

- H2: **Thingworx-Platform-H2-7.2.0**
- PostgreSQL/HA: **Thingworx-Platform-Postgres-7.2.0**

High Availability Option

With ThingWorx 7.0 and later, you can use PostgreSQL with an optional High Availability layer at the database level and/or at the ThingWorx level. Additional steps for HA are required and are located in the ThingWorx High Availability Administrator's Guide.

For detailed software and hardware requirements, refer to the ThingWorx System Requirements and Compatibility Matrix document.

Installing ThingWorx for the First Time: H2 or PostgreSQL on Windows

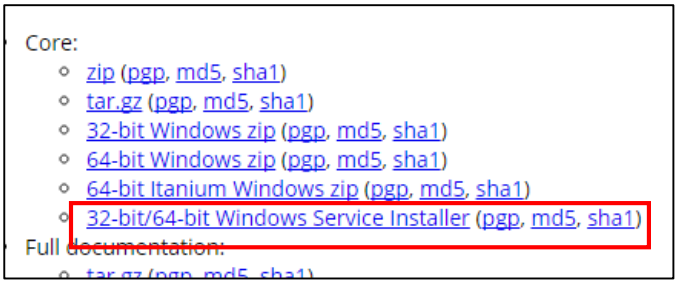
NOTE: If you are using PostgreSQL, additional steps are required. If you are installing H2, steps are included below to skip all PostgreSQL sections.

Installing Oracle Java and Apache Tomcat (Windows)

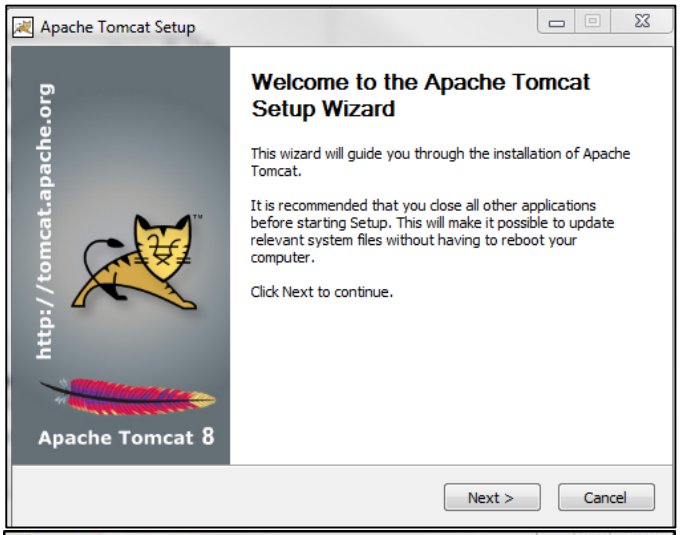
1. Download and install the required version of Java from the Oracle website.
   NOTE: Refer to the System Requirements and Compatibility Matrix document for version requirements.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

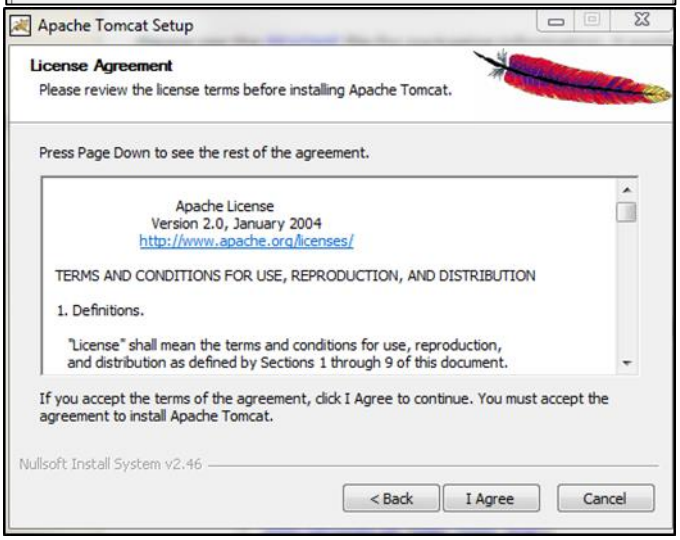
2. Visit the Tomcat website to download the **32-bit/64-bit Windows Service Installer (pgp, md5, sha1).**
   NOTE: Refer to the System Requirements and Compatibility Matrix document for version requirements.



3. The Apache Tomcat Setup Wizard launches. Click **Next**.

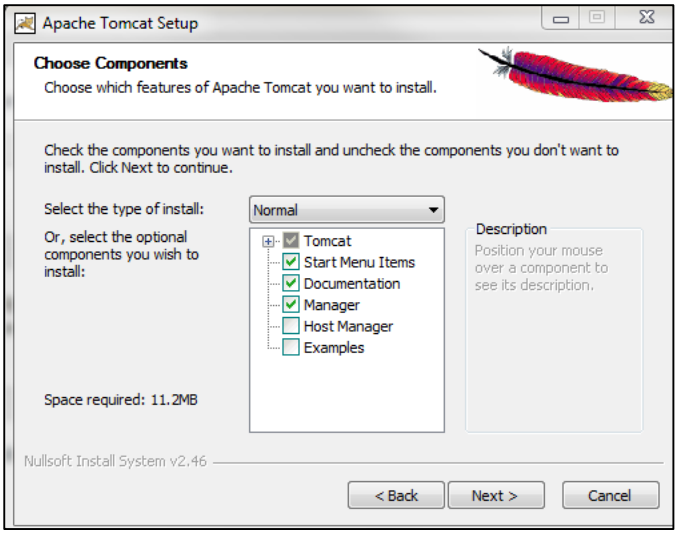


4. Click **I Agree**.

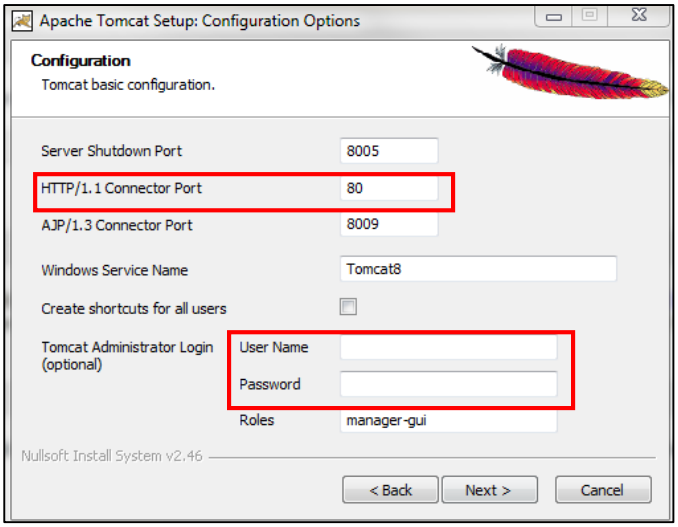


Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

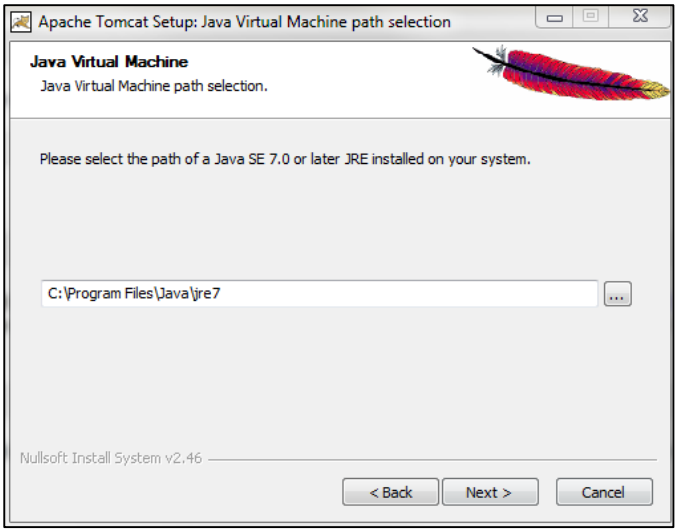
5. In the **Components** section, use the default settings.
6. Click **Next**.



7. In the **HTTP/1.1 Connector Port** field, type **80 (or other available port)**.
8. In the **Tomcat Administrator Login** fields, type a **User Name** and **Password**.
9. Click **Next**.

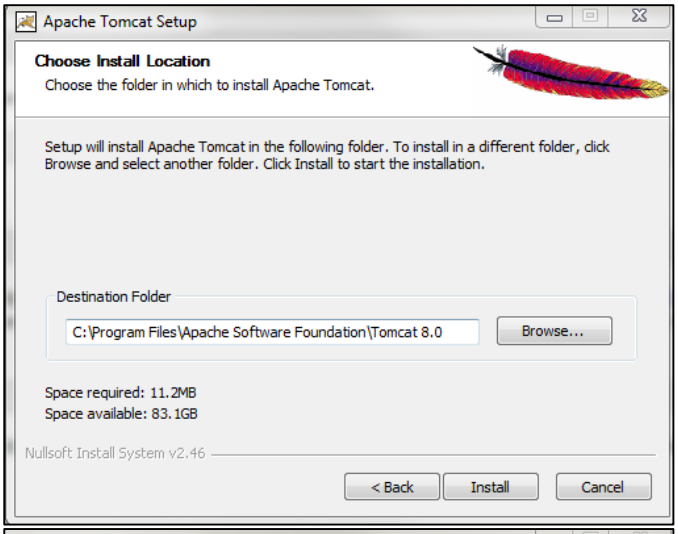


10. Enter the path to the proper 64-bit Java installation directory.
11. Click **Next**.

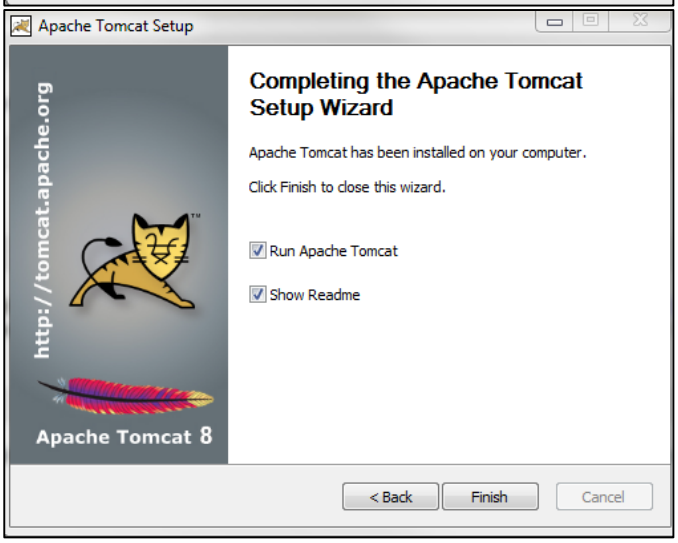


Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

12. Click **Install**.



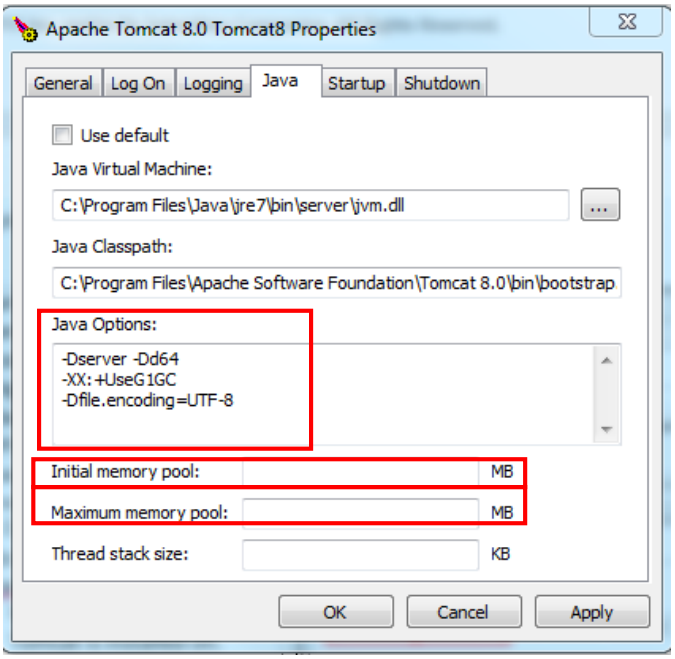
13. Click **Finish**.



Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

14. Click **Start>Configure Tomcat**.
15. Open the **Java** tab.
16. In the Java Options field, add the following to the end of the options field:
    -Dserver -Dd64
    -XX:+UseG1GC
    -Dfile.encoding=UTF-8

    NOTE: For more information on these options and for additional options for hosted and/or public-facing environments, refer to the Appendix: Tomcat Java Option Settings.
17. Clear any values in the **Initial memory pool** and **Maximum memory pool** fields.
18. Click **OK**.



19. Go to the location of the Tomcat installation and open the **server.xml** file in the **conf** folder. For example, C:\Program Files\Apache Software Foundation\Tomcat 8.0\conf\server.xml
20. Replace **HTTP/1.1** with **protocol="org.apache.coyote.http11.Http11NioProtocol"**
21. Save and close the file.

```
    <!--
    <Connector port="8443"
protocol="org.apache.coyote.http11.Http11NioProtocol"
               maxThreads="150" SSLEnabled="true" scheme="https"
secure="true"
               clientAuth="false" sslProtocol="TLS" />
    -->
```

22. OPTIONAL STEP: If you want to increase the default cache settings that affect static file caching, add the following line within the <context></context> tags in the $TOMCAT_HOME/conf/context.xml file:
    ```
    <Resources
    cacheMaxSize="501200"
    cacheObjectMaxSize="2048"
    cacheTtl="60000"/>
    ```

23. If you are installing H2, skip to the Installing ThingWorx section.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Installing and Configuring PostgreSQL (Windows)

The instructions provided below are intended for the PostgreSQL administrator (not the DB host servers).

NOTE: If you are including the HA layer to your implementation, refer to the ThingWorx High Availability Administrator's Guide.

This section includes the following:

- Installing PostgreSQL
- Creating a new user role in PostgreSQL
- Configuring and executing the PostgreSQL database script (thingworxPostgresDBSetup.bat)
- Configuring and executing the model/data provider schema script (thingworxPostgresSchemaSetup.bat)
- Configuring platform-settings.json

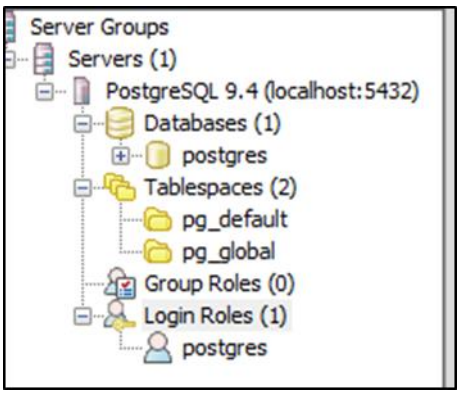
Installing PostgreSQL and Creating a New User Role in PostgreSQL (Windows)

1. Download and install the appropriate version of PostgreSQL from the following site: http://www.postgresql.org/download/

- pgAdmin III Tool
  - PgAdmin III is an open source management tool for your databases that is included in the PostgreSQL download. The tool features full Unicode support, fast, multithreaded query, and data editing tools and support for all PostgreSQL object types.

2. Open PostgreSQL using pgAdmin III.

3. Create a new user role (in this example, it is **twadmin**):
   a. Right click **PostgreSQL9.4** (localhost:5432).
   b. Select **NewObject>New Login Role**. On the **Properties** tab, in the **Role name** field, type **twadmin**.
   c. On the **Definition** tab, in the **Password** field, type **password** (must type twice).



4. Click **OK**.
   NOTE: Remember the user role

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

name created in this step for later use.

Configuring and Executing the PostgreSQL Database Script (Windows)

To set up the PostgreSQL database and tablespace, the **thingworxPostgresDBSetup.bat** script must be configured and executed.

1. Add the **<postgres-installation>/bin** folder to your system path variable.
2. Create a directory named **ThingworxPostgresqlStorage** on your Thingworx Storage drive.

   NOTE: If you create with the -**d<databasename>**, you do not have to use the postgres user.

   NOTE: You must specify the **-l** option to a path that exists. For example, **-l D:\ThingworxPostgresqlStorage**. The script does not create the folder for you.

   The folder must have appropriate ownership and access rights. It should be owned by the same user who runs the PostgreSQL service, and have Full Control assigned to that user - this user is generally **NETWORK_SERVICE**, but may differ in your environment.

3. Obtain and open **thingworxPostgresDBSetup.bat** from the ThingWorx software download package.
4. Configure the script. Reference the configuration options in the table below.

   Various parameters such as **server**, **port**, **database**, **tablespace**, **tablespace location** and **thingworxusername** can be configured in the script, depending on the requirements. Execute this script with the **--help** option for usage information.

   As an example, to set up the database and tablespace with a default Postgres installation that has a postgres database as well as a postgres user name and assuming the user created above is **twadmin**, enter:

   thingworxpostgresDBSetup –a postgres –u twadmin –l C:\ThingworxPostgresqlStorage

   where **twadmin** is the user name

5. Execute the script. Once executed, this creates a new database and tablespace with a default PostgreSQL installation in the PostgreSQL instance installed on the localhost.

   NOTE: You may need to run the command prompt as admin.

thingworxPostgresDBSetup.bat Script Options

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Option	Parameter	Default	Description	Example
-t or -T	server	localhost	Tablespace name	-t thingworx
-p or -P	port	5432	Port number of PostgreSQL	-p 5432
-d or -D	database	thingworx	PostgreSQL Database name to create	-d thingworx
-h or -H	tablespace	thingworx	Name of the PostgreSQL tablespace.	-h localhost
-l or -L	tablespace_location	/ThingworxPostgresqlStorage	Required. Location in the file system where the files representing database objects are stored.	-l or -L
-a or -A	adminusername	postgres	Administrator Name	-a postgres
-u or -U	thingworxusername	twadmin	User name that has permissions to write to the database.	-u twadmin

Configuring and Executing the Model/Data Provider Schema Script (Windows)

To set up the PostgreSQL model/data provider schema, the **thingworxPostgresSchemaSetup.bat** script must be configured and executed. This will set up the public schema under your database on the PostgreSQL instance installed on the localhost.

1. Obtain and open the **thingworxPostgresSchemaSetup.bat** from the ThingWorx software download package.
2. Configure the script. Reference the configuration options in the table below.

   Various parameters such as **server**, **port**, **database**, **username**, **schema**, and **option** can be configured in the script depending on the requirements. Execute this script with **--help** option for usage information.

3. Execute the script.
   NOTE: You may be prompted to provide your password three times.

thingworxPostgresSchemaSetup.bat Script Options

Option	Parameter	Default	Description	Example
-h or -H	server	localhost	IP or host name of the database	-h localhost

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Option	Parameter	Default	Description	Example
-p or -P	port	5432	Port number of PostgreSQL	-p 5432
-d or -D	database	thingworx	Database name to use	-d thingworx
-s or -S	schema	public	Schema name to use	-s mySchema
-u or -U	username	twadmin	Username to update the database schema	-u twadmin
-o or -O	option	all	There are three options: **all**: Sets up the model and data provider schemas into the specified database. **model**: Sets up the model provider schema into the specified database. **data**: Sets up the data provider schema into the specified database.	-o data

Configuring platform-settings.json (Windows)

1. To use the default ThingworxPlatform configuration directory, create a folder called **ThingworxPlatform** at the root of the drive where Tomcat was installed. Alternatively, if you want to specify the location where ThingWorx stores its settings, you can set the **THINGWORX_PLATFORM_SETTINGS** environment variable to the desired location.

   Ensure that the folder referenced by **THINGWORX_PLATFORM_SETTINGS** exists and is writable by the Tomcat user. This environment variable should be configured as part of the system environment variables.
2. Create a file named **platform-settings.json** and place the file into the **ThingworxPlatform** folder.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

3. Open **platform-settings.json** and configure as necessary. Refer to the configuration options in Appendix C: platform-settings.json Options and Appendix B: platform-settings.json sample.

NOTE: If your PostgreSQL server is not the same as your ThingWorx server, and you are having issues with your ThingWorx installation, review your Tomcat logs and platform-settings.json file. The default installation assumes both servers are on the same machine.

Encrypting the PostgreSQL Password (Windows)

If you want to provide added security encryption for the PostgreSQL database settings in the platform-settings.json file, you can do the following.

Note: This encryption process is optional.

Prerequisites

- You must have Java installed and on your path.
- You must have PostgreSQL installed and know the password.

1. Create a working directory such as C:\password_setup (windows), and copy the Thingworx.war there.
2. Unzip the Thingworx.war.
3. Open a command prompt, cd to your working directory, and set your CLASSPATH by doing the following:
   a. Go to Control Panel > System Properties > Environment Variables.
   b. Create a new environment variable: PG_PW_UTIL
      `C:\password_setup\WEB-INF\lib\thingworx-platform-common-<release-version>.jar;C:\password_setup\WEB-INF\lib\slf4j-api-1.7.12.jar;C:\password_setup\WEB-INF\lib\logback-core-1.0.13.jar;C:\password_setup\WEB-INF\lib\logback-classic-1.0.13.jar;C:\password_setup\WEB-INF\lib\thingworx-common-<release-version>.jar`
   c. Add the variable to the CLASSPATH.
      `CLASSPATH`
      `<don't touch existing classpath>; %PG_PW_UTIL%`
   d. In your command shell, enter `'java -version'`.
      It should respond with a Java version.
4. Open /ThingworxPlatform/platform-settings.json and change the password value to `'encrypt.db.password'`.
   For example, `"password": "encrypt.db.password",`
5. Create a directory named `"\twx"` that has the same parent directory as the `"\ThingworxPlatform"` directory.
6. To create a key store with the PostgreSQL password encrypted inside, run the following command: `java com.thingworx.platform.security.keystore.ThingworxKeyStore encrypt.db.password <postgres_password>`
   The second argument must be your PostgreSQL password.
7. Once you have created the encrypted password, remove the updates to the CLASSPATH.

Installing ThingWorx (Windows)

1. Locate the appropriate **Thingworx.war** file.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

NOTE: ThingWorx downloads are available in PTC Software Downloads.

2. Copy the **Thingworx.war** file and place it in the following location of your Tomcat installation:
   \Apache Software Foundation\Tomcat 8.0\webapps

3. To launch ThingWorx, go to **<servername>/Thingworx** in a web browser.
   NOTE: Use a strong password. The login information below is for the Administrator user only.

   Use the following login information:
   Login Name: Administrator
   Password: admin

Installing ThingWorx for the First Time: PostgreSQL or H2 on Ubuntu

Oracle Java, and Apache Tomcat, and PostgreSQL(if using PostgreSQL) must be installed prior to installing ThingWorx. Refer to the System Requirements and Compatibility Matrix for specific version requirements.

Installing Oracle Java and Apache Tomcat (Ubuntu)

1. Update Ubuntu packages:

   ```
   $ sudo apt-get update
   ```

2. Install and Configure Network Time Protocol (NTP) settings for time synchronization:

   ```
   $ sudo apt-get install ntp
   ```

   NOTE: The default configuration for NTP is sufficient. For additional configuration information about NTP (beyond the scope of this documentation), refer to the following resources:
   - Time Synchronization with NTP
   - How do I use pool.ntp.org?

3. Edit AUTHBIND properties to allow Tomcat to bind to ports below 1024:

   ```
   $ sudo apt-get install authbind
   ```

4. Download the Java (JDK) tar file from Oracle's website and upload it to the server using scp or sftp.

   NOTE: Refer to the System Requirements and Compatibility Matrix document for specific version requirements.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

5. Extract tar file:

```
$ tar -xf jdk-8u45-linux-x64.tar.gz
```

6. Create the directory by moving the JDK to `/usr/lib/jvm`:

```
$ sudo mkdir -p /usr/lib/jvm
$ sudo mv jdk1.8.0_45/ /usr/lib/jvm/
```

7. Add alternatives to the system:

```
$ sudo update-alternatives --install "/usr/bin/java" "java"
"/usr/lib/jvm/jdk1.8.0_45/bin/java" 1
$ sudo update-alternatives --install "/usr/bin/keytool"
"keytool" "/usr/lib/jvm/jdk1.8.0_45/bin/keytool" 1
```

8. Change access permissions:

```
$ sudo chmod a+x /usr/bin/java

$ sudo chmod a+x /usr/bin/keytool
```

9. Change owner:

```
$ sudo chown -R root:root /usr/lib/jvm/jdk1.8.0_45/
```

10. Configure master links:

```
$ sudo update-alternatives --config java
$ sudo update-alternatives --config keytool
```

NOTE: Additional executables in /usr/lib/jvm/jdk1.8.0_45/bin/ can be installed using the previous set of steps.

11. Verify Java version:

```
$ java -version
java version "1.8.0_45"
Java(TM) SE Runtime Environment (build 1.8.0_45-b14)
Java HotSpot(TM) 64-Bit Server VM (build 24.75-b04, mixed mode)
```

12. Download Apache Tomcat:

```
$ wget http://archive.apache.org/dist/tomcat/tomcat-
8/v8.0.33/bin/apache-tomcat-8.0.33.tar.gz
```

13. Extract tar file:

```
$ tar -xf apache-tomcat-8.0.33.tar.gz
```

14. Move Tomcat to `/usr/share/tomcat8`:

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

```
$ sudo mkdir -p /usr/share/tomcat8
$ sudo mv apache-tomcat-8.0.33 /usr/share/tomcat8/8.0.33
```

15. Define environment variables in `/etc/environment`:

    ```
    export JAVA_HOME=/usr/lib/jvm/jdk1.8.0_45
    export CATALINA_HOME=/usr/share/tomcat8/8.0.33
    ```

    NOTE: `/etc/environment` is read at boot, so a reboot is necessary.

16. Change directory to $CATALINA_HOME:

    ```
    $ cd $CATALINA_HOME
    ```

17. Add user and group to the system:

    ```
    $ sudo addgroup --system tomcat8 --quiet
    $ sudo adduser --system --home /usr/share/tomcat8/ --no-create-
    home --ingroup tomcat8 --disabled-password --shell /bin/false
    tomcat8
    ```

18. Change owner and access permissions of `bin/ lib/` and `webapps/` :

    ```
    $ sudo chown -Rh tomcat8:tomcat8 bin/ lib/ webapps/
    $ sudo chmod 775 bin/ lib/ webapps/
    ```

19. Change owner and access permissions of `conf/`:

    ```
    $ sudo chown -Rh root:tomcat8 conf/
    $ sudo chmod 640 conf/*
    ```

20. Change access permissions of `logs/, temp/,` and `work/`:

    ```
    $ sudo chown -R tomcat8:adm logs/ temp/ work/
    $ sudo chmod 750 logs/ temp/ work/
    ```

21. In `bin/`, create `setenv.sh` with the following contents:

    ```
    # Java Options
    export JAVA_OPTS="-Djava.awt.headless=true -
    Djava.net.preferIPv4Stack=true -Dserver -Dd64 -XX:+UseNUMA -
    XX:+UseConcMarkSweepGC -Dfile.encoding=UTF-8"
    export JRE_HOME=/usr/lib/jvm/jdk1.8.0_45/jre
    ```

    NOTE: For more information on these options and for additional options for hosted and/or public-facing environments, refer to the Appendix: Tomcat Java Option Settings.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

22. Change owner and access permissions of `bin/setenv.sh`:

```
$ sudo chown tomcat8:tomcat8 bin/setenv.sh
$ sudo chmod 775 bin/setenv.sh
```

23. Create self-signed certificate:

```
$ $JAVA_HOME/bin/keytool -genkey -alias tomcat8 -keyalg RSA
$ sudo cp ~/.keystore $CATALINA_HOME/conf/
$ sudo chown root:tomcat8 $CATALINA_HOME/conf/.keystore
$ sudo chmod 640 $CATALINA_HOME/conf/.keystore
```

24. Uncomment the Manager element in context.xml to prevent sessions from persisting across restarts:

```
<Manager pathname="" />
```

25. Modify the shutdown string and protocol used by the SSL Connector in `conf/server.xml` (comment out the non-SSL Connector):

```
<Server port="8005" shutdown="TH!nGW0rX">

<Connector port="443"
protocol="org.apache.coyote.http11.Http11NioProtocol"
maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
keystoreFile="${user.home}/8.0.33/conf/.keystore"
keystorePass="changeit" clientAuth="false" sslProtocol="TLS" />
```

26. Define a user in `conf/tomcat-users.xml`:

```
<user username="tomcat" password="tomcat" roles="manager"/>
```

    NOTE: In hosted and/or public-facing environments, use of the manager web application is not recommended because it introduces a security risk. Similarly, the example web applications included in `/webapps` should be removed as they may introduce unnecessary security vulnerabilities into Tomcat.

27. Determine id of tomcat8 user:

```
$ id -u tomcat8
117
```

28. In `/etc/authbind/byuid/`, create id file (for example, 117) with the following contents:

```
0.0.0.0/0:1,1023
```

29. Change owner and access permissions of `/etc/authbind/byuid/117`:

```
$ sudo chown tomcat8:tomcat8 /etc/authbind/byuid/117
$ sudo chmod 700 /etc/authbind/byuid/117
```

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

30. Modify `bin/startup.sh` to always use authbind:

```
#exec "$PRGDIR"/"$EXECUTABLE" start "$@"
exec authbind --deep "$PRGDIR"/"$EXECUTABLE" start "$@"
```

31. In `/etc/init.d`, create tomcat8 file with the following contents:

```
CATALINA_HOME=/usr/share/tomcat8/8.0.33

case $1 in
  start)
    /bin/su -p -s /bin/sh tomcat8 $CATALINA_HOME/bin/startup.sh
  ;;

  stop)
    /bin/su -p -s /bin/sh tomcat8 $CATALINA_HOME/bin/shutdown.sh
  ;;

  restart)
    /bin/su -p -s /bin/sh tomcat8 $CATALINA_HOME/bin/shutdown.sh
    /bin/su -p -s /bin/sh tomcat8 $CATALINA_HOME/bin/startup.sh
  ;;

esac
exit 0
```

32. Change access permissions of `etc/init.d/tomcat8` and create symbolic links:

```
$ sudo chmod 755 /etc/init.d/tomcat8
$ sudo ln -s /etc/init.d/tomcat8 /etc/rc1.d/K99tomcat
$ sudo ln -s /etc/init.d/tomcat8 /etc/rc2.d/S99tomcat
```

33. OPTIONAL: If you want to increase the default cache settings that affect static file caching, add the following line within the <context></context> tags in the $TOMCAT_HOME/conf/context.xml file:

```
<Resources cacheMaxSize="501200" cacheObjectMaxSize="2048"
cacheTtl="60000"/>
```

34. If you are installing H2, skip to the Installing ThingWorx section.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Installing and Configuring PostgreSQL (Ubuntu)

The instructions provided below are intended for the PostgreSQL administrator (not the DB host servers).

NOTE: If you are including the HA layer to your implementation, refer to the ThingWorx High Availability Administrator's Guide.

This section includes the following:

- Installing PostgreSQL
- Creating a new user role in PostgreSQL
- Configuring and executing the PostgreSQL database script (thingworxPostgresDBSetup.bat)
- Configuring and executing the model/data provider schema script (thingworxPostgresSchemaSetup.bat)
- Configuring platform-settings.json

Installing PostgreSQL and Creating a New User Role in PostgreSQL (Ubuntu)

1. Download and install the appropriate version of PostgreSQL.

   In a Ubuntu environment, this can be installed directly from the package manager:

   sudo apt-get install postgresql-9.4

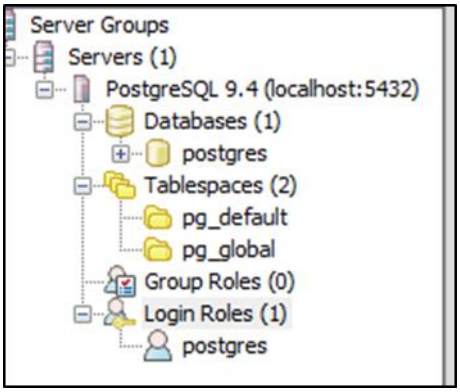
2. Open PostgreSQL using pgAdmin III. In a Ubuntu 15.10 environment, it can be installed directly from the package manager:

   sudo apt-get install pgadmin3

- pgAdmin III Tool
  - PgAdmin III is an open source management tool for your databases that is included in the PostgreSQL download. The tool features full Unicode support, fast, multithreaded query, and data editing tools and support for all PostgreSQL object types.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

3. Create a new user role (in this example, it is **twadmin**):
    a. Right click **PostgreSQL9.4** (localhost:5432).
    b. Select **NewObject>New Login Role**. On the **Properties** tab, in the **Role name** field, type **twadmin**.
    c. On the **Definition** tab, in the **Password** field, type **password** (must type twice).
4. Click **OK**.
   NOTE: Remember the user role name created in this step for later use.



Configuring and Executing the PostgreSQL Database Script (Ubuntu)

To set up the PostgreSQL database and tablespace, the **thingworxPostgresDBSetup.sh** script must be configured and executed.

1. Add the **<postgres-installation>/bin** folder to your system path variable.
2. Create a directory named **ThingworxPostgresqlStorage** on your Thingworx Storage drive.

   NOTE: You must specify the **-l** option to a path that exists. For example, **-l D:\ThingworxPostgresqlStorage**. The script does not create the folder for you.
   The folder needs have appropriate ownership and access rights. It should be owned by the postgres user and have the read, write, and execute assigned to the owner.

   NOTE: If you create with the -**d<databasename>**, you do not have to use the postgres user.

3. Obtain and open **thingworxPostgresDBSetup.sh** from the ThingWorx software download package.
4. Configure the script. Reference the configuration options in the table below.

   Various parameters such as **server**, **port**, **database**, **tablespace**, **tablespace location** and **thingworxusername** can be configured in the script, depending on the requirements. Execute this script with the **--help** option for usage information.

   As an example, to set up the database and tablespace with a default Postgres installation that has a postgres database as well as a postgres user name and assuming the user created above is **twadmin**, enter:

   thingworxpostgresDBSetup –a postgres –u twadmin –l C:\ThingworxPostgresqlStorage

   where **twadmin** is the user name

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

5. Execute the script. Once executed, this creates a new database and tablespace with a default PostgreSQL installation in the PostgreSQL instance installed on the localhost.

thingworxPostgresDBSetup.sh Script Options

Option	Parameter	Default	Description	Example
-t or -T	server	localhost	Tablespace name	-t thingworx
-p or -P	port	5432	Port number of PostgreSQL	-p 5432
-d or -D	database	thingworx	PostgreSQL Database name to create	-d thingworx
-h or -H	tablespace	thingworx	Name of the PostgreSQL tablespace.	-h localhost
-l or -L	tablespace_location	/ThingworxPostgresqlStorage	Required. Location in the file system where the files representing database objects are stored. *	-l or -L
-a or -A	adminusername	postgres	Administrator Name	-a postgres
-u or -U	thingworxusername	twadmin	User name that has permissions to write to the database.	-u twadmin

Configuring and Executing the Model/Data Provider Schema Script (Ubuntu)

To set up the PostgreSQL model/data provider schema, the **thingworxPostgresSchemaSetup.sh** script must be configured and executed. This will set up the public schema under your database on the PostgreSQL instance installed on the localhost.

1. Obtain and open the **thingworxPostgresSchemaSetup.sh** from the ThingWorx software download package.
2. Configure the script. Reference the configuration options in the table below.

   Various parameters such as **server**, **port**, **database**, **username**, **schema**, and **option** can be configured in the script depending on the requirements. Execute this script with **--help** option for usage information.

3. Execute the script.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

thingworxPostgresSchemaSetup.sh Script Options

Option	Parameter	Default	Description	Example
-h or -H	server	localhost	IP or host name of the database	-h localhost
-p or -P	port	5432	Port number of PostgreSQL	-p 5432
-d or -D	database	thingworx	Database name to use	-d thingworx
-s or -S	schema	public	Schema name to use	-s mySchema
-u or -U	username	twadmin	Username to update the database schema	-u twadmin
-o or -O	option	all	There are three options: **all**: Sets up the model and data provider schemas into the specified database. **model**: Sets up the model provider schema into the specified database. **data**: Sets up the data provider schema into the specified database.	-o data

Configuring platform-settings.json (Ubuntu)

1. Create a folder called **ThingworxPlatform** at the root (for example, **\ThingworxPlatform** or as a system variable (for example, **THINGWORX_PLATFORM_SETTINGS=/data/ThingworxPlatform**).
2. Create a file named **platform-settings.json** and place the file into the **ThingworxPlatform** folder.
3. Configure as necessary. Refer to the configuration options in Appendix C: platform-settings.json Options and Appendix B: platform-settings.json sample.

NOTE: If your PostgreSQL server is not the same as your ThingWorx server, and you are having issues with your ThingWorx installation, review your Tomcat logs and platform-settings.json file. The default installation assumes both servers are on the same machine.

Encrypting the PostgreSQL Password (Ubuntu)

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

If you want to provide added security encryption for the PostgreSQL database settings in the platform-settings.json file, you can do the following.
Note: This encryption process is optional.

Prerequisites

- You must have Java installed and on your path.
- You must have PostgreSQL installed and know the password.

1. Create a working directory such as ~/password_setup, and copy the Thingworx.war there.
2. Unzip the Thingworx.war.
3. Open a command prompt, cd to your working directory, and set your CLASSPATH to the following:
   `export CLASSPATH=WEB-INF/lib/thingworx-platform-common-<release-version>.jar:WEB-INF/lib/slf4j-api-1.7.12.jar:WEB-INF/lib/logback-core-1.0.13.jar:WEB-INF/lib/logback-classic-1.0.13.jar:./WEB-INF/lib/thingworx-common-<release-version>.jar`
4. Open /ThingworxPlatform/platform-settings.json and change the password value to `'encrypt.db.password'`.
   For example, `"password": "encrypt.db.password",`
5. Create a directory named `"/twx"` that has the same parent directory as `"/ThingworxPlatform"`:
   `sudo mkdir /twx`
6. To create a key store with the PostgreSQL password encrypted inside, run the following command: `java com.thingworx.platform.security.keystore.ThingworxKeyStore encrypt.db.password <postgres_password>`
   The second argument must be your PostgreSQL password.
7. Run the following commands:
   ```
   sudo chown tomcat8:tomcat8 /twx
   sudo chmod 755 /twx
   ```

Installing ThingWorx (Ubuntu)

1. Create `/ThingworxStorage` and `/ThingworxBackupStorage` directories:

   ```
   $ sudo mkdir /ThingworxStorage /ThingworxBackupStorage
   ```

2. Change owner and access permissions of `/ThingworxStorage` and `/ThingworxBackupStorage`:

   ```
   $ sudo chown tomcat8:tomcat8 /ThingworxStorage
   /ThingworxBackupStorage
   $ sudo chmod 775 /ThingworxStorage /ThingworxBackupStorage
   ```

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

3. Unzip the ThingWorx zip archive and move to `$CATALINA_HOME/webapps`:

```
$ unzip Thingworx-Platform-7.1.0.latest.zip (or appropriate
version)
$ sudo mv Thingworx.war $CATALINA_HOME/webapps
$ sudo chown tomcat8:tomcat8
$CATALINA_HOME/webapps/Thingworx.war
$ sudo chmod 775 $CATALINA_HOME/webapps/Thingworx.war
```

4. Start Tomcat to deploy the ThingWorx web application:

```
$ sudo service tomcat8 start
```

NOTE: Use a strong password. The login information below is for the Administrator user only.

Username: **Administrator**
Password: **admin**

Installing and Configuring ThingWorx for the First Time: PostgreSQL or H2 on Red Hat Enterprise Linux (RHEL)

Oracle Java, and Apache Tomcat, and PostgreSQL (if using PostgreSQL) must be installed prior to installing ThingWorx.

Installing Oracle Java and Apache Tomcat (RHEL)

1. Download the Java installer. Open a terminal and run:

wget --no-check-certificate -c --header "Cookie: oraclelicense=accept-securebackup-cookie" http://download.oracle.com/otn-pub/java/jdk/8u45-b14/jdk-8u45-linux-x64.rpm

2. Run the Java installer:

sudo rpm -i jdk-8u45-linux-x64.rpm

3. Create the directory and move the JDK:

sudo mkdir -p /usr/lib/jvm
sudo mv /usr/java/jdk1.8.0_45/ /usr/lib/jvm/

4. Set the Java alternatives:

sudo alternatives --install /usr/bin/java java /usr/lib/jvm/jdk1.8.0_60/bin/java 1

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

sudo alternatives --install /usr/bin/keytool keytool /usr/lib/jvm/jdk1.8.0_45/bin/keytool

5. Change access permissions:

   sudo chmod a+x /usr/bin/java
   sudo chmod a+x /usr/bin/keytool

6. Change Owner:

   sudo chown -R root:root /usr/lib/jvm/jdk1.8.0_45/

7. Configure master links:

   sudo alternatives --config java
   ∗ select the option that contains /usr/lib/jvm/jdk1.8.0_45/bin/java
   sudo ln -s /usr/lib/jvm/jdk1.8.0_45/bin/keytool /usr/bin/keytool
   sudo alternatives --config keytool

8. Verify Java version:

   java –version
   java version "1.8.0_45"
   Java(TM) SE Runtime Environment (build 1.8.0_45-b14)
   Java HotSpot(TM) 64-Bit Server VM (build 25.45-b02, mixed mode)

9. Install Tomcat. Download the Tomcat installer:

   wget https://archive.apache.org/dist/tomcat/tomcat-8/v8.0.33/bin/apache-tomcat-8.0.33.tar.gz

10. Extract the contents:

    tar -xf apache-tomcat-8.0.33.tar.gz

11. Move Tomcat to /usr/share/tomcat8:

    sudo mkdir -p /usr/share/tomcat8
    sudo mv apache-tomcat-8.0.33 /usr/share/tomcat8/8.0.33

12. Change directory to /usr/share/tomcat8/8.0.33:

    cd /usr/share/tomcat8/8.0.33

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

13. Add user and group to the system:
    sudo groupadd -r tomcat8
    sudo useradd -r -d /usr/share/tomcat8 -g tomcat8 -s /bin/false tomcat8

14. Change owner and access permissions of bin/ lib/ and webapps/

    sudo chown -Rh tomcat8:tomcat8 bin/ lib/ webapps/
    sudo chmod 775 bin/ lib/ webapps/

15. Change owner and access permissions of conf/:

    sudo chown -Rh root:tomcat8 conf/
    sudo chmod 640 conf/*

16. Change access permissions of logs/, temp/, and work/:

    sudo chown -R tomcat8:adm logs/ temp/ work/
    sudo chmod 750 logs/ temp/ work/

17. Create a bin/setenv.sh file and paste these contents:

    sudo touch bin/setenv.sh

    Open bin/setenv.sh in an editor (as root), paste the following and save:

    ```
    # Java Options
    export JAVA_OPTS="-Djava.awt.headless=true -
    Djava.net.preferIPv4Stack=true -Dserver -Dd64 -
    XX:+UseNUMA -XX:+UseConcMarkSweepGC -
    Dfile.encoding=UTF-8"
    export JRE_HOME=/usr/lib/jvm/jdk1.8.0_45/jre
    ```

    NOTE: For more information on these options and for additional options for hosted and/or public-facing environments, refer to the Appendix: Tomcat Java Option Settings.

18. Change owner and access permissions of bin/setenv.sh:

    sudo chown tomcat8:tomcat8 bin/setenv.sh
    sudo chmod 775 bin/setenv.sh

19. Create self-signed certificate:

    /usr/lib/jvm/jdk1.8.0_45/jre/bin/keytool -genkey -alias tomcat8 -keyalg RSA

    Follow the instructions to complete the certificate creation process.
    Set the keystore password to **changeit**

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Set the tomcat8 user password to the same as the keystore password

sudo cp ~/.keystore /usr/share/tomcat8/8.0.33/conf/
sudo chown root:tomcat8 /usr/share/tomcat8/8.0.33/conf/.keystore
sudo chmod 640 /usr/share/tomcat8/8.0.33/conf/.keystore

20. Uncomment the Manager element in context.xml to prevent sessions from persisting across restarts.

    Open **/usr/share/tomcat8/8.0.33/conf/context.xml** in a text editor (as root) and remove the '`<!—`' before '`<Manager pathname="" />`' and the '`-->`' after

21. Save the file.

22. Modify the shutdown string and protocol used by the SSL Connector in server.xml:

    Open **/usr/share/tomcat8/8.0.33/conf/server.xml** in a text editor (as root)

    Change '`<Server port="8005" shutdown="SHUTDOWN">`' to '`<Server port="8005" shutdown="TH!nGW0rX ">`'

    Comment out or remove this section:

    ```
    ' <Connector port="8080" protocol="HTTP/1.1"
    connectionTimeout="20000" redirectPort="8443" />'
    ```

    Paste in this section directly below:

    ```
    <Connector port="443"
    protocol="org.apache.coyote.http11.Http11NioProto
    col"
    maxThreads="150" SSLEnabled="true" scheme="https"
    secure="true"
       keystoreFile="${user.home}/8.0.33/conf/.keysto
       re" keystorePass="changeit" clientAuth="false"
       sslProtocol="TLS" />
    ```

23. Save the file.

24. Define an Apache Manager user in tomcat-users.xml:

    Open **/usr/share/tomcat8/8.0.33/conf/tomcat-users.xml** in a text editor (as root)
    Just above the final line (</tomcat-users>) add the following line:

    ```
    <user username="tomcat" password="tomcat"
    ```

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

```
roles="manager,manager-gui"/>
```

25. Save the file.
    NOTE: The roles included are for ease of testing and can be removed if security is a concern.

26. Set up Tomcat as a service to start on boot. First, build JSVC:

    sudo yum install gcc
    sudo rm /usr/java/latest
    sudo ln -s /usr/lib/jvm/jdk1.8.0_45 /usr/java/latest
    cd /usr/share/tomcat8/8.0.33/bin/
    sudo tar xvfz commons-daemon-native.tar.gz
    cd commons-daemon-*-native-src/unix
    sudo ./configure --with-java=/usr/java/latest
    sudo make
    sudo cp jsvc ../..

27. Create the Tomcat service file:

    sudo touch /usr/lib/systemd/system/tomcat.service

    Open **/usr/lib/systemd/system/tomcat.service** in a text editor (as root) and paste in the following:

```
[Unit]
Description=Apache Tomcat Web Application Container
After=network.target

[Service]
Type=forking
PIDFile=/var/run/tomcat.pid
Environment=CATALINA_PID=/var/run/tomcat.pid
Environment=JAVA_HOME=/usr/lib/jvm/jdk1.8.0_45
Environment=CATALINA_HOME=/usr/share/tomcat8/8.0.33
Environment=CATALINA_BASE=/usr/share/tomcat8/8.0.33
Environment=CATALINA_OPTS=

ExecStart=/usr/share/tomcat8/8.0.33/bin/jsvc \
                -Dcatalina.home=${CATALINA_HOME} \
                -Dcatalina.base=${CATALINA_BASE} \
                -cp ${CATALINA_HOME}/bin/commons-
daemon.jar:${CATALINA_HOME}/bin/bootstrap.jar:${CATALINA_
HOME}/bin/tomcat-juli.jar \
                -user tomcat8 \
                -java-home ${JAVA_HOME} \
                -pidfile /var/run/tomcat.pid \
                -errfile
```

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

```
                    ${CATALINA_HOME}/logs/catalina.out \
                    -outfile
                    ${CATALINA_HOME}/logs/catalina.out \
                    $CATALINA_OPTS \
                    org.apache.catalina.startup.Bootstrap

ExecStop=/usr/share/tomcat8/8.0.33/bin/jsvc \
                    -pidfile /var/run/tomcat.pid \
                    -stop \
                    org.apache.catalina.startup.Bootstrap

[Install]
WantedBy=multi-user.target
```

28. Set Tomcat to run on system start up:

    sudo systemctl enable tomcat.service

    Note: This will allow the user to control the Tomcat service with the following commands:

    systemctl start tomcat
    systemctl stop tomcat
    systemctl restart tomcat
    systemctl status tomcat

29. Use the firewall to redirect port 80 to the secure Tomcat port:

    sudo firewall-cmd --zone=public --add-forward-port=port=80:proto=tcp:toport=443 –permanent
    sudo firewall-cmd –reload

30. Start the Tomcat service and test:

    sudo systemctl start tomcat

    You should now be able to connect to the Tomcat server by entering **https://localhost** in a browser.

31. If you are installing H2, skip to the Installing ThingWorx section.

Installing and Configuring PostgreSQL (RHEL)

The instructions provided below are intended for the PostgreSQL administrator (not the DB host servers).

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

NOTE: If you are including the HA layer to your implementation, refer to the ThingWorx High Availability Administrator's Guide.

This section includes the following:

- Installing PostgreSQL
- Creating a new user role in PostgreSQL
- Configuring and executing the PostgreSQL database script (thingworxPostgresDBSetup.bat)
- Configuring and executing the model/data provider schema script (thingworxPostgresSchemaSetup.bat)
- Configuring platform-settings.json

Installing PostgreSQL and Creating a New User Role in PostgreSQL (RHEL)

1. Add the PostgreSQL repository to Yum and install:

   rpm -Uvh http://yum.postgresql.org/9.4/redhat/rhel-7-x86_64/pgdg-redhat94-9.4-5.noarch.rpm
   sudo yum install postgresql94 postgresql94-server postgresql94-contrib

2. Install PgAdmin III:

   sudo yum install pgadmin3

3. Initialize and launch the database:

   sudo /usr/pgsql-9.4/bin/postgresql94-setup initdb

4. Set the PostgreSQL service to start on boot:

   sudo chkconfig postgresql-9.4 on
   sudo service postgresql-9.4 start

5. Set up password for the postgres user:
   sudo passwd postgres

   Enter the password for the postgres user
   Take note of this password.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

6. Set up postgres user in psql:

   sudo -u postgres psql -c "ALTER ROLE postgres WITH password 'password'"

   The password should be the same as in the step above.

7. Configure pgadmin3.

   sudo pgadmin3

   * In the pgAdminIII GUI, click on **file->Open postgresql.conf**

   * Open **/var/lib/pgsql/9.4/data/postgresql.conf**
   * Put a check next to **listen addresses** and **port**
     -The default settings of "localhost" and "5432" are usually sufficient.
   * Save and close.
   * Click on **file->Open pg_hba.conf**
   * Open **/var/lib/pgsql/9.4/data/pg_hba.conf**
   * Double-click on the line with address **127.0.0.1/32**
   * Set Method to **md5**
   * Double-click on the line with address **1/128**
   * Set Method to **md5**
   * Click **OK**
   * Save and exit
   * Close pgadmin3

8. Restart the PostgreSQL service:

   sudo service postgresql-9.4 restart

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

9. Set up pgadmin3 to connect to the database.

   sudo pgadmin3

   Click the plug **Add a connection to a server** in the top left corner.
   Fill out the following:
   ```
   Name: PostgreSQL 9.4
   Host: localhost
   Port: 5432
   Service: <blank>
   Maintenance DB: postgres
   Username: postgres
   Password: <password as set in step above>
   Store password: Checked
   Group: ServersLocalhost
   ```
   Click **OK**

10. Create a new user role (in this example, it is **twadmin**):

    Right click **PostgreSQL9.4 (localhost:5432**).
    Note: It may be possible to activate some extensions. Click **Databases** and select **postgres** in the main window. A dialog displays. Click **Fix it!**
    Select **NewObject>New Login Role**.
    On the **Properties** tab, in the **Role name** field, type **twadmin**.
    On the **Definition** tab, in the **Password** field, type password (must type twice).

11. Click **OK**.

12. Create the ThingworxPostgresqlStorage directory:

    sudo mkdir /ThingworxPostgresqlStorage
    sudo chmod 775 /ThingworxPostgresqlStorage
    sudo chown postgres:postgres /ThingworxPostgresqlStorage/
    sudo mkdir /ThingworxPlatform
    sudo chmod 775 /ThingworxPlatform
    sudo chown tomcat8:tomcat8 /ThingworxPlatform

13. Download the ThingWorx installer from the PTC downloads page:
    NOTE: The file used in this example is **ThingWorx-Platform-Postgres-6-6-2 (MED-61111-CD-066_SP2_ThingWorx-Platform-Postgres-6-6-2.zip).**

    mkdir ~/Thingworx
    cp MED-61111-CD-066_SP2_ThingWorx-Platform-Postgres-6-6-2.zip ~/Thingworx/
    cd ~/Thingworx
    unzip MED-61111-CD-066_SP2_ThingWorx-Platform-Postgres-6-6-2.zip

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

14. Execute the PostgreSQL Database Script:

    cd install
    sudo sh thingworxPostgresDBSetup.sh -a postgres -u twadmin -l /ThingworxPostgresqlStorage

15. Execute the Model/Data Provider Schema Script:

    sh thingworxPostgresSchemaSetup.sh

    NOTE: When prompted, use the password for twadmin that was previously set up.

16. Startup configuration of platform-settings.json:

    sudo cp ~/Thingworx/platform-settings.json /ThingworxPlatform/

NOTE: If your PostgreSQL server is not the same as your ThingWorx server, and you are having issues with your ThingWorx installation, review your Tomcat logs and platform-settings.json file. The default installation assumes both servers are on the same machine.

Encrypting the PostgreSQL Password (RHEL)

If you want to provide added security encryption for the PostgreSQL database settings in the platform-settings.json file, you can do the following.
Note: This encryption process is optional.

Prerequisites

- You must have Java installed and on your path.
- You must have PostgreSQL installed and know the password.

1. Create a working directory such as ~/password_setup, and copy the Thingworx.war there.
2. Unzip the Thingworx.war.
3. Open a command prompt, cd to your working directory, and set your CLASSPATH to the following:
   `export CLASSPATH=WEB-INF/lib/thingworx-platform-common-<release-version>.jar:WEB-INF/lib/slf4j-api-1.7.12.jar:WEB-INF/lib/logback-core-1.0.13.jar:WEB-INF/lib/logback-classic-1.0.13.jar:./WEB-INF/lib/thingworx-common-<release-version>.jar`
4. Open /ThingworxPlatform/platform-settings.json and change the password value to `'encrypt.db.password'`.
   For example, `"password": "encrypt.db.password",`
5. Create a directory named `"/twx"` that has the same parent directory as `"/ThingworxPlatform"`:
   `sudo mkdir /twx`
6. To create a key store with the PostgreSQL password encrypted inside, run the following command: `java com.thingworx.platform.security.keystore.ThingworxKeyStore`

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

```
encrypt.db.password <postgres_password>
```
The second argument must be your PostgreSQL password.

7. Run the following commands:
```
sudo chown tomcat8:tomcat8 /twx
sudo chmod 755 /twx
```

Installing ThingWorx (RHEL)

1. Create `/ThingworxStorage` and `/ThingworxBackupStorage` directories:

   $ sudo mkdir /ThingworxStorage /ThingworxBackupStorage

2. Change owner and access permissions of `/ThingworxStorage` and `/ThingworxBackupStorage`:

   $ sudo chown tomcat8:tomcat8 /ThingworxStorage/ThingworxBackupStorage
   $ sudo chmod 775 /ThingworxStorage /ThingworxBackupStorage

3. Move Thingworx.war to Tomcat/webapps

   sudo mv ~/Thingworx/Thingworx.war /usr/share/tomcat8/8.0.33/webapps/.
   sudo chown tomcat8:tomcat8 /usr/share/tomcat8/8.0.33/webapps/Thingworx.war
   udo chmod 775 /usr/share/tomcat8/8.0.33/webapps/Thingworx.war

4. Restart Tomcat to start ThingWorx:

   sudo systemctl restart tomcat

5. Log into ThingWorx Composer:

   In a browser, open **https://localhost/Thingworx/Composer/index.html**

   NOTE: Use a strong password. The login information below is for the Administrator user only.
   User: **Administrator**
   Password: **admin**

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

NOTE: If you are performing an in-place migration, the following step is not necessary.

1. Import extensions. In Compser, click **Import/Export>Import**.

   NOTE: Obtain and import the latest versions of the extensions. If you are upgrading to a major version (for example, from 6.x to 7.0, you must import the 7.x versions of the extensions.)
   Extensions are available in PTC Software Downloads and the ThingWorx Marketplace.

   NOTE: For in-place migration from 6.5 to 7.0 for Neo4j with DataStax Enterprise (DSE), an additional Tomcat restart is required when you are installing the latest version of:
   - DsePersistenceProvider_ExtensionPackage.zip
     NOTE: This extension must be requested from Support.

2. For in-place migration from 6.5 to 7.0 for Neo4j/PostgreSQLwith DSE ONLY: Additional steps are required after importing the DsePersistenceProvider_ExtensionPackage.zip extension.

   NOTE: This extension must be requested from Support.

   Neo4j with DSE: A “bulkInsertException” validation error may display when the extension is initially imported, but restarting Tomcat will clear the error.
   PostgreSQL with DSE: An import error will display when the extension is initially imported. Restart Tomcat and reimport the extension.

NOTE: If you are performing an in-place migration, the following step is not necessary.

3. Import entities and data. In Composer, click **Import/Export>From ThingworxStorage**.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Appendix A: Tomcat Java Option Settings

Mandatory Settings

Setting	Description
-server	Explicitly tells the JVM to run in server mode. This is true by default when using 64-bit JDK, but it is best practice to declare it.
-d64	Explicitly tells the JVM to run in 64-bit mode. The current JVM automatically detects this, but it is best practice to declare it.
XX:+UseG1GC	Tells the JVM to use the Garbage First Garbage Collector.
-Dfile.encoding=UTF-8	Tells the JVM to use UTF-8 as the default character set so that non-Western alphabets are displayed correctly.
-Xms3072m (for a system with 4GB of memory)	Tells the JVM to allocate a minimum of 3072MB of memory to the Tomcat process. This should be set to 75% of the available system memory. NOTE: The amount of memory needs to be tuned depending on the actual environment.
-Xmx3072m (for a system with 4GB of memory)	Tells the JVM to limit the maximum memory to the Tomcat process. This should be set to 75% of the available system memory. NOTE: The amount of memory needs to be tuned depending on the actual environment. 5GB of memory is a good starting point for 100,000 things. NOTE: The reason that the min and max amounts of memory are made equal is to reduce JVM having to re-evaluate required memory and resizing the allocation at runtime. While this is recommended for hosted and/or public-facing environments, for development and test environments, using **–Xms512m** would suffice. Also, verify that there is enough memory left to allow the operating system to function.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Optional Settings to Enable JMX Monitoring for VisualVM or JConsole

Setting	Description
-Dcom.sun.management.jmxremote	Notifies the JVM that you plan to remote monitor it via JMX
-Dcom.sun.management.jmxremote.port=22222	The port the JVM should open up for monitoring.
-Dcom.sun.management.jmxremote.ssl=false	No SSL usage.
-Dcom.sun.management.jmxremote.authenticate=false	No authentication required.
-Djava.rmi.server.hostname=<host or IP>	The hostname or IP that the underlying RMI client connection will use.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Appendix B: Sample platform-settings.json

```
{
      "PlatformSettingsConfig": {
            "BasicSettings": {
                  "BackupStorage": "/ThingworxBackupStorage",
                  "DatabaseLogRetentionPolicy": 7,
                  "EnableBackup": true,
                  "EnableHA": false,
                  "EnableSystemLogging": false,
                  "HTTPRequestHeaderMaxLength": 2000,
                  "HTTPRequestParameterMaxLength": 2000,
                  "Storage": "/ThingworxStorage"
            },

            "HASettings": {
                  "CoordinatorConnectionTimeout": 15000,
                  "CoordinatorHosts": "127.0.0.1:2181",
                  "CoordinatorMaxRetries": 3,
                  "CoordinatorRetryTimeout": 1000,
                  "CoordinatorSessionTimeout": 60000,
                  "LoadBalancerBase64EncodedCredentials":
"QWRtaW5pc3RyYXRvcjphZG1pbg=="
            }
      },

      "PersistenceProviderPackageConfigs": {

            "H2PersistenceProviderPackage": {
                  "ConnectionInformation": {
                        "acquireIncrement": 5,
                        "acquireRetryAttempts": 30,
                        "acquireRetryDelay": 1000,
                        "checkoutTimeout": 2000,
                        "idleConnectionTestPeriod": 6,
                        "initialPoolSize": 10,
                        "maxConnectionAge": 0,
                        "maxIdleTime": 0,
                        "maxIdleTimeExcessConnections": 36000,
                        "maxPoolSize": 100,
                        "maxStatements": 0,
                        "maxStatementsPerConnection": 50,
                        "minPoolSize": 10,
```

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

```
                "numHelperThreads": 6,
                "tableLockTimeout": 10000,
                "testConnectionOnCheckout": false,
                "unreturnedConnectionTimeout": 0
        },

        "StreamProcessorSettings": {
                "maximumBlockSize": 2500,
                "maximumQueueSize": 250000,
                "maximumWaitTime": 10000,
                "numberOfProcessingThreads": 5,
                "scanRate": 5,
                "sizeThreshold": 1000
        },

        "ValueStreamProcessorSettings": {
                "maximumBlockSize": 2500,
                "maximumWaitTime": 10000,
                "maximumQueueSize": 500000,
                "numberOfProcessingThreads": 5,
                "scanRate": 5,
                "sizeThreshold": 1000
        }
},
"NeoPersistenceProviderPackage": {
        "StreamProcessorSettings": {
                "maximumBlockSize": 2500,
                "maximumQueueSize": 250000,
                "maximumWaitTime": 10000,
                "scanRate": 5,
                "sizeThreshold": 1000
        },

        "ValueStreamProcessorSettings": {
                "maximumBlockSize": 2500,
                "maximumQueueSize": 500000,
                "maximumWaitTime": 10000,
                "scanRate": 5,
                "sizeThreshold": 1000
        }
},

"PostgresPersistenceProviderPackage": {
```

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

```
                "ConnectionInformation": {
                      "acquireIncrement": 5,
                      "acquireRetryAttempts": 3,
                      "acquireRetryDelay": 10000,
                      "checkoutTimeout": 1000000,
                      "driverClass": "org.postgresql.Driver",
                      "fetchSize": 5000,
                      "idleConnectionTestPeriod": 60,
                      "initialPoolSize": 5,
                      "jdbcUrl":
"jdbc:postgresql://localhost:5432/thingworx",
                      "maxConnectionAge": 0,
                      "maxIdleTime": 0,
                      "maxIdleTimeExcessConnections": 300,
                      "maxPoolSize": 100,
                      "maxStatements": 100,
                      "minPoolSize": 5,
                      "numHelperThreads": 8,
                      "password": "password",
                      "testConnectionOnCheckout": false,
                      "unreturnedConnectionTimeout": 0,
                      "username": "twadmin"
                },

                "StreamProcessorSettings": {
                      "maximumBlockSize": 2500,
                      "maximumQueueSize": 250000,
                      "maximumWaitTime": 10000,
                      "numberOfProcessingThreads": 5,
                      "scanRate": 5,
                      "sizeThreshold": 1000
                },

                "ValueStreamProcessorSettings": {
                      "maximumBlockSize": 2500,
                      "maximumQueueSize": 500000,
                      "maximumWaitTime": 10000,
                      "numberOfProcessingThreads": 5,
                      "scanRate": 5,
                      "sizeThreshold": 1000
                }
          }
     }
```

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

```
}
```

Appendix C: platform-settings.json Options

NOTE: The platform-settings.json file is optional for H2. It is available for administrators to adjust settings for fine-tuning.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

platform-settings.json Options		
Setting	Default	Description
Core Platform Settings		
BackupStorage	/ThingworxBackupStorage	The directory name where all backups are written to.
DatabaseLogRetentionPolicy	7	The number of days that database logs are retained.
EnableBackup	true	Determines whether backups are retained.
EnableHA	false	Determines whether the PlatformThingWorx Core can be configured for a highly available landscape.
EnableSystemLogging	false	Determines whether system logging is enabled. NOTE: DO NOT TURN THIS ON UNLESS INSTRUCTED BY THINGWORX SUPPORT.
HTTPRequestHeaderMaxLength	2000	The maximum allowable length for HTTP Request Headers values.
HTTPRequestParameterMaxLength	2000	The maximum allowable length for HTTP Request Parameter values.
Storage	/ThingworxStorage	The directory where all storage directories are created/located (excluding Backup Storage).
HA Settings Settings specific to a PostgreSQL HA landscape configuration. All are optional, and are ignored if the **EnableHA** setting above is set to **false**.		
CoordinatorConnectionTimeout	15000	How long to wait (in milliseconds) for a connection to be established with process/server used to coordinate PlatformThingWorx Core leadership.
CoordinatorHosts	127.0.0.1:2181	A comma-delimited list of server IP addresses on which the processes used to coordinate PlatformThingWorx Core leadership exist (e.g. "127.0.0.1:2181, 127.0.0.2:2181").
CoordinatorMaxRetries	3	The maximum allowable number of retries that will be made to establish a connection with the process/server used to coordinate PlatformThingWorx Core leadership.
CoordinatorRetryTimeout	1000	How long to wait (in milliseconds) for each retry attempt.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

CoordinatorSessionTimeout	60000	How long the Platform's ThingWorx Core session is to wait (in milliseconds) without receiving a "heartbeat" from the process/server used to coordinate PlatformThingWorx Core leadership.
LoadBalancerBase64EncodedCredentials	QWRtaW5pc3RyYXRvcjphZG1pbg=="	The Base64-encoded credentials for the HA Load Balancer, in the format of *<user>:<password>*. NOTE: You can use any utility that Base64 encodes the matching *<user>:<password>* string used in your load balancer setup.
PostgresPersistenceProviderPackage PostgreSQL-specific persistence provider settings. If PostgreSQL is not the persistence provider, then this entire section should be ignored.		
acquireIncrement	5	Determines how many connections at a time the platform will try to acquire when the pool is exhausted.
acquireRetryAttempts	3	Defines how many times the PlatformThingWorx Core will try to acquire a new Connection from the database before giving up.
acquireRetryDelay	10000	The time (in milliseconds) the PlatformThingWorx Core will wait between acquire attempts.
checkoutTimeout	10000000	The number of milliseconds a client calling getConnection() will wait for a Connection to be checked-in or acquired when the pool is exhausted.
driverClass	org.postgresql.Driver	The fully-qualified class name of the JDBC driverClass that is expected to provide Connections.
fetchSize	5000	The count of rows to be fetched in batches instead of caching all rows on the client side.
idleConnectionTestPeriod	60	If this is a number greater than 0, the PlatformThingWorx Core will test all idle, pooled but unchecked-out connections, every x number of seconds.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

initialPoolSize	5	Initial number of database connections created and maintained within a pool upon startup. Should be between minPoolSize and maxPoolSize.
jdbcUrl	jdbc:postgresql://localhost:5432/thingworx”	The jdbc url used to connect to PostgreSQL. NOTE: If the default schema name is changed (from public), you must add <databasename>?currentSchema=<name of schema> For example, if the schema name is mySchema, it would be: **jdbc:postgresql://<*DBServer*>:<*DBPort*>/<*databasename*>?currentSchema=mySchema** NOTE: If you are configuring an HA solution, this should reflect the server IP that the pgPool process is running on. Change the port to the port that pgPool is serving.
maxConnectionAge	0	Seconds, effectively a time to live. A Connection older than maxConnectionAge will be destroyed and purged from the pool.
maxIdleTime	0	Seconds a connection can remain pooled but unused before being discarded. Zero means idle connections never expire.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

maxIdleTimeExcessConnections	300	The number of seconds that connections in excess of minPoolSize are permitted to remain in idle in the pool before being culled. Intended for applications that wish to aggressively minimize the number of open connections, shrinking the pool back towards minPoolSize if, following a spike, the load level diminishes and Connections acquired are no longer needed. If maxIdleTime is set, maxIdleTimeExcessConnections should be smaller to have any effect. Setting this to zero means no enforcement and excess connections are not idled out.
maxPoolSize	100	Maximum number of Connections a pool will maintain at any given time.
maxStatements	100	The size of the PlatformThingWorx Core's global PreparedStatement cache.
minPoolSize	5	Minimum number of Connections a pool will maintain at any given time.
numHelperThreads	8	The number of helper threads to spawn. Slow JDBC operations are generally performed by helper threads that don't hold contended locks. Spreading these operations over multiple threads can significantly improve performance by allowing multiple operations to be performed simultaneously.
password	password	The password used to log into the database.
testConnectionOnCheckout	false	If true, an operation will be performed at every connection checkout to verify that the connection is valid.
unreturnedConnectionTimeout	0	The number of seconds to wait for a response from an unresponsive connection before discarding it. If set, if an application checks out but then fails to check-in a connection within the specified period of time, the pool will discard the connection.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

		This permits applications with occasional connection leaks to survive, rather than eventually exhausting the Connection pool. Zero means no timeout, and applications are expected to close their own connections.
username	twadmin	The user that has the privilege to modify tables. This is the user created on the database for the ThingWorx server.
Stream Processor Settings		
maximumBlockSize	2500	The maximum number of stream writes to process in one block.
maximumQueueSize	250000	The maximum number of stream entries to queue (will be rejected after that)
maximumWaitTime	10000	Number of milliseconds the system waits before flushing the stream buffer.
numberOfProcessingThreads	5	The number of processing threads (cannot change for Neo4j).
scanRate	5	The buffer status is checked at the specified rate value in milliseconds.
sizeThreshold	1000	Maximum number of items to accumulate before flushing the stream buffer.
Value Stream Processor Settings		
maximumBlockSize	2500	Maximum number of value stream writes to process in one block.
maximumQueueSize	500000	Maximum number of value stream entries to queue (will be rejected after that).
maximumWaitTime	10000	Number of milliseconds the system waits before flushing the value stream buffer.
numberofProcessingThreads	5	The number of processing threads (cannot change for Neo4j).
scanRate	5	The rate (in milliseconds) before flushing the stream buffer.
sizeThreshold	1000	Maximum number of items to accumulate before flushing the value stream buffer.
NeoPersistenceProviderPackage		

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Contains Neo4j-specific Persistence Provider settings. If Neo is not the Persistence Provider, then this entire section should be ignored.		
StreamProcessorSettings		
maximumBlockSize	2500	The maximum number of stream writes to process in one block.
maximumQueueSize	250000	The maximum number of stream entries to queue (will be rejected after that).
maximumWaitTime	10000	The maximum wait time (in milliseconds) before flushing stream buffer.
scanRate	5	The rate (in milliseconds) at which to check the buffer status.
sizeThreshold	1000	The maximum number of items to accumulate before flushing stream buffer.
V**alueStreamProcessorSettings**		
maximumBlockSize	2500	The maximum number of stream writes to process in one block.
maximumQueueSize	500000	The maximum number of stream entries to queue (will be rejected after that).
maximumWaitTime	10000	The maximum wait time (in milliseconds) before flushing the stream buffer.
scanRate	5	The rate (in milliseconds) at which to check the buffer status.
sizeThreshold	1000	The maximum number of items to accumulate before flushing stream buffer.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

Appendix D: Metrics Reporting

You can opt in to allow your ThingWorx Core metrics data (such as usage, performance, and diagnostics) to be sent to a PTC server. The configuration settings for metrics reporting are included in the Platform Subsystem. For more information, see the Platform Subsystem topic in the Help Center.

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.