



Upgrading to ThingWorx 7.2

Version 1.0

Copyright © 2016 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
July 28, 2016	1.0	Initial version for 7.2



Installing ThingWorx

Document Revision History..... 1

 Database Options: Neo4j, PostgreSQL, or H2..... 3

 Schema Changes 3

 Hot Fix Process to Upgrading..... 3

 Prerequisites 3

 Upgrading to ThingWorx 7.2 (Neo4j/PostgreSQL/H2) on Windows..... 4

 Upgrading to ThingWorx 7.2 (Neo4j/PostgreSQL/H2) on Ubuntu 10

Appendix: Data/Entity Migration Scenarios 14

 In-place Migrations 14

Upgrading ThingWorx

This guide contains steps for upgrading to a newer version of ThingWorx. If you are installing ThingWorx for the first time, refer to the [Installing ThingWorx](#) guide.

Refer to [Appendix: Data/Entity Migration Scenarios](#) before performing the steps in this guide. This will determine if you can perform an in-place migration.

If you are installing ThingWorx for the first time, refer to the [Installing ThingWorx](#) guide.

Database Options: Neo4j, PostgreSQL, or H2

With ThingWorx 7.2, you can use PostgreSQL (with an optional High Availability layer), Neo4j, 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**
- Neo4j: **Thingworx-Platform-Neo-7.2.0**
- PostgreSQL/HA: **Thingworx-Platform-Postgres-7.2.0**

Schema Changes

There are two possible scenarios when upgrading ThingWorx. Additional steps to back up entities and data are required if you are upgrading when there are schema updates.

- Upgrading when there is a schema update
 - a. When upgrading from a beta version to an official release, you should assume a schema change is required.
 - b. When you change minor versions (for example, 7.0 to 7.2) you should assume a schema change.
- Upgrading when there is not a schema update
 - a. Upgrading to a service pack level (for example, 7.2.0 to 7.2.1) typically does not require a schema change.

Hot Fix Process to Upgrading

If you are working in a development environment, do not have any data or entities to migrate, and are using the same database provider (Neo4j or PostgreSQL), you can use the following steps to get started quickly.

1. Undeploy the thingworx.war.
2. Restart Tomcat.
3. Deploy the new thingworx.war.

Prerequisites

- Before upgrading, be sure to back up your model and runtime data.

- If you have localization tables with locales that are not in the `language-Script-REGION-variant` format, you should rename them before migrating by doing the following:
 1. From the ThingWorx Explorer, select **System ► Localization Tables**.
A list of localization tables appears.
 2. Select a localization table with a non-conforming name (for example, *French*).
 3. Choose **Duplicate**.
 4. Enter the standard locale name (in this example for French, enter *fr*).
 5. Choose **Save**.
 6. Delete the original localization table with the non-conforming name.
 7. Repeat the above steps for all tables with names that do not conform to the standard.

After importing the localization tables into your new system, do the following:

1. From the ThingWorx Explorer, select **System ► Localization Tables**.
2. Select a localization table to edit.
3. Enter values in the **Language Name (Native)** field (for example, *français*) and **Language Name (Common)** field (for example, enter *French*).
4. If you want to choose or change an avatar (such as a national flag), click Change next to the Avatar field.
5. The avatar appears in the language preferences editor.
6. Repeat these steps for all imported localization tables.

Upgrading to ThingWorx 7.2 (Neo4j/PostgreSQL/H2) on Windows

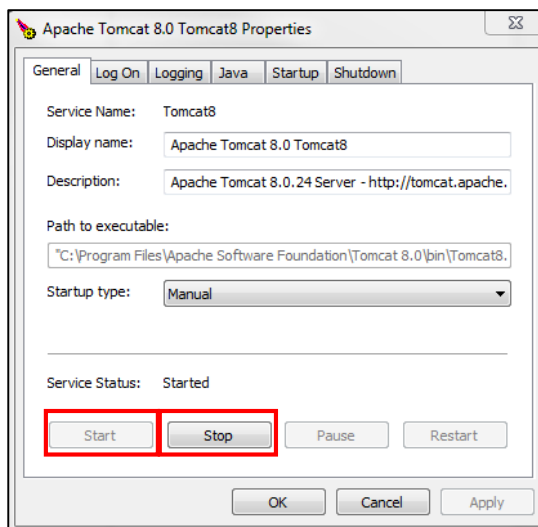
NOTE: If you have data and entities to migrate, refer to [Appendix B: Data/Entity Migration Scenarios and Best Practices](#) for recommendations regarding your scenario. If you are using PostgreSQL, it is recommended to backup/take a snapshot of the database and all tables prior to performing the migration steps below. Some steps in this section may be omitted based on your scenario.

1. Obtain the latest version of ThingWorx.

NOTE: ThingWorx downloads are available in [PTC Software Downloads](#).

2. Verify that you are running the required versions of Tomcat and Oracle.
 - Refer to the [System Requirements and Compatibility Matrix](#) document for version requirements.

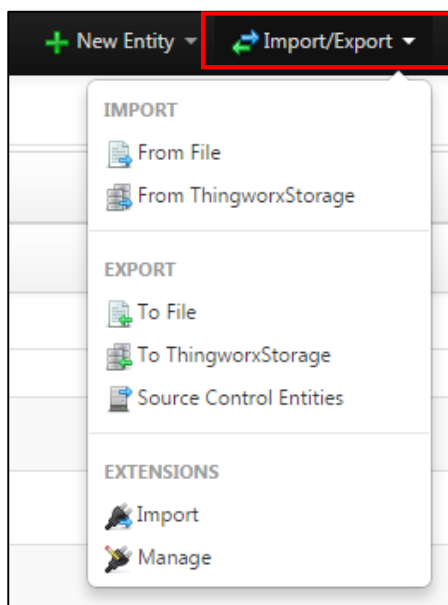
3. Restart Tomcat: In the Tomcat Properties, click **Stop**. Wait for Tomcat to stop.
4. It is highly recommended that you backup the following two folders before continuing:
 - Apache Software Foundation/Tomcat 8.0/webapps/Thingworx
 - <drive>:\ThingworxStorage
Where <drive> is the drive Tomcat is installed on.
5. If you are upgrading from 6.0 or above, remove the **validation.properties** file from /ThingworxStorage/esapi
- NOTE: If you have custom configurations in the file, move the file for later reference.
6. Click **Start**.
Restarting Tomcat assures that the database is clear before exporting.



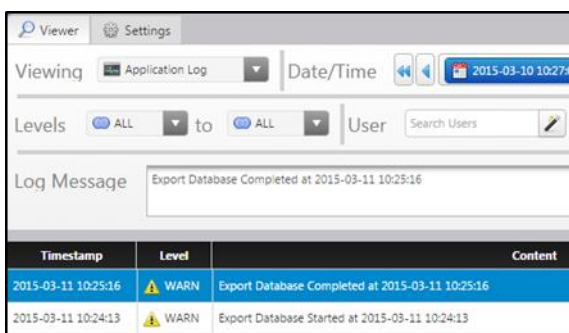
NOTE: The **validation.properties** file is created upon startup of ThingWorx. If you do not remove the file, the updated file with additional parameters will not overwrite the current version during upgrade. Reference the [Help Center](#) for additional information.

7. Export entities and data* (if necessary). In Composer, click **Import/Export>To ThingworxStorage**.

*For in-place migration using DSE (Neo4j or PostgreSQL): If you have any data located in Neo4j and you are persisting any blog, wiki, stream, value stream, or data table data to DSE, do not export all data to **ThingworxStorage** (to prevent duplicating blog, wiki, stream, value stream, or data table data when imported). Instead, you must explicitly export Neo4j data to file.

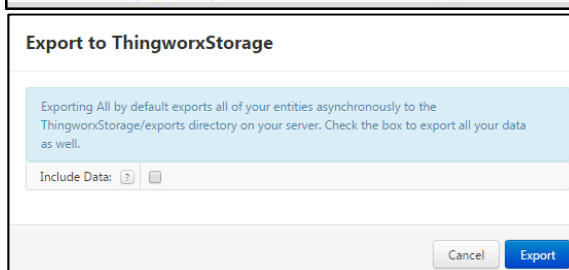


NOTE: Data export progress can be monitored in the Application Log.



8. If necessary, click **Include Data**.
9. Click **Export**.

NOTE: Data and entities are exported to ThingworxStorage/exports



10. Copy these data and entity export files and move to a safe location. You will import these files in a later step.

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

11. Note any extensions that are in use.

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

12. Delete the **ThingworxStorage** and **ThingworxBackupStorage** folders.

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

13. Move the exportfiles from step 10 back to the **ThingworxStorage\exports** folder.
14. Go to the Tomcat installation at **\Apache Software Foundation\Tomcat 8.0\webapps** and delete the **Thingworx.war** file.
15. If you are performing an in-place migration from PostgreSQL 6.5 or above to 7.2, run the following scripts that are located in the **update** folder (starting with the version you are upgrading from):
 - **thingworxPostgresSchemaUpdate6.5-to-6.6.bat**
 - **thingworxPostgresSchemaUpdate6.6-to-7.0.bat**
 - **thingworxPostgresSchemaUpdate7.0-to-7.1.bat**
 - **thingworxPostgresSchemaUpdate7.1-to-7.2.bat**
16. Copy the **Thingworx.war** file and place it in the following location of your Tomcat installation:
\Apache Software Foundation\Tomcat 8.0\webapps

17. 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

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

18. If necessary, import any extensions.

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.x, you must import the 7.x versions of the extensions.) Extensions are available in [PTC Software Downloads](#) and the [ThingWorx Marketplace](#).

19. For in-place migration from 6.5 to 7.0 for Neo4j/PostgreSQL with DSE ONLY: Additional steps are required after importing the **DsePersistenceProvider_ExtensionPackage.zip** extension.
NOTE: This extension must be requested from Support.

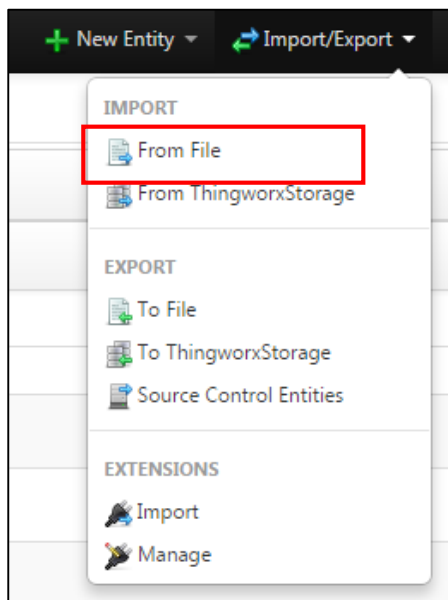
Neo4j with DSE: A

“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.

20. Import entities and data. In Composer, click **Import/Export> From File**.



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

21. Select the data and/or entities to import.
 - a. Select the **Use Default Persistence Provider** check box if your data/entities were exported from 6.0. See the note to the right.
 - b. Select the **Ignore Subsystems** checkbox if you want to ignore the Subsystem settings of the imported entities (for example, if you are going from QA to production).

NOTE on Upgrading 6.0 Neo4j to 6.5 PostgreSQL only

In ThingWorx 6.0, the default persistence provider was named NeoPersistenceProvider and any blog, wiki, data table, stream, or value stream used this as its default persistence provider. In 6.5, the default persistence provider name was changed to ThingworxPersistenceProvider to make it easier to port between Platforms of different model providers. If you're currently using the NeoPersistenceProvider in any of these entity types, you'll need to check the **Use Default Persistence Provider** option in the Import dialog box so that the import will automatically switch to the correct default persistence provider. Otherwise, the import will fail with an error message that reads:

Error importing: error, Import Failed: Thing Thing: TestStream had an invalid persistence provider [NeoPersistenceProvider] assigned

Upgrading to ThingWorx 7.2 (Neo4j/PostgreSQL/H2) on Ubuntu

NOTE: If you have data and entities to migrate, refer to [Appendix B: Data/Entity Migration Scenarios](#) for recommendations regarding your scenario. If you are using PostgreSQL, it is recommended to backup/take a snapshot of the database and all tables prior to performing the migration steps below. Some steps in this section may be omitted based on your scenario.

1. Obtain the latest version of ThingWorx.
NOTE: ThingWorx downloads are available in [PTC Software Downloads](#).
2. Restart Tomcat:

```
$ sudo service tomcat8 stop
$ sudo service tomcat8 start
```

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

*For in-place migration using DSE (Neo4j or PostgreSQL): If you have any data located in Neo4j and you are persisting any blog, wiki, stream, value stream, or data table data to DSE, do not export all data to **ThingworxStorage** (to prevent duplicating blog, wiki, stream, value stream, or data table data when imported). Instead, you must explicitly export Neo4j data to file.

NOTE: Data export progress can be monitored in the Application Log.

4. If necessary, click **Include Data**.

5. Click **Export**.

NOTE: Data and entities are exported to **ThingworxStorage/exports**.

6. Copy the ThingworxStorage and ThingworxBackupStorage folders to a temporary location for later retrieval.
7. It is highly recommended that you backup the contents of the following two folders before continuing:
 - **Apache Software Foundation/Tomcat8.0/webapps/Thingworx**
 - **/ThingworxStorage**

8. Remove the **validation.properties** file from **/ThingworxStorage/esapi**
NOTE: If you have custom configurations in the file, move the file for later reference.

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

9. Note any extensions that are in use (located in **/ThingworxStorage/extensions**).

10. Delete the **/ThingworxStorage** and **/ThingworxBackupStorage** folders.

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

11. Move the exports file back to the **ThingworxStorage\exports** folder.

12. Remove the contents of the following folder:
Apache Software Foundation/Tomcat8.0/webapps/Thingworx.

13. Start Tomcat.

```
$ sudo service tomcat8 start
```

14. Undeploy ThingWorx.

15. If you are performing an in-place migration from PostgreSQL 6.5 or later to 7.2, run the following scripts that are located in the **update** folder (starting with the version you are upgrading from):

- **thingworxPostgresSchemaUpdate6.5-to-6.6.sh**
- **thingworxPostgresSchemaUpdate6.6-to-7.0.sh**
- **thingworxPostgresSchemaUpdate7.0-to-7.1.sh**
- **thingworxPostgresSchemaUpdate7.1-to-7.2.sh**

16. If you are upgrading to 6.6 PostgreSQL, locate your **ThingworxPlatform** directory and place the **platform-settings.json** file with the configuration from the desired build.

NOTE: If you do not have a **ThingworxPlatform** directory; create it in the root directory of the drive you used to install Tomcat.

17. Locate the appropriate Thingworx.war file.

NOTE: ThingWorx downloads are available in [PTC Software Downloads](#).

18. Unzip the ThingWorx zip archive and move to \$CATALINA_HOME/webapps:

```
$ unzip Thingworx-Platform-7.2.0.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
```

19. Start Tomcat to deploy the ThingWorx web application:

```
$ sudo service tomcat8 start
```

20. 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

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

21. Import extensions. In Composer, 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 7.2, 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 7.2 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.

22. For in-place migration from 6.5 to 7.2 for Neo4j/PostgreSQL with 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.

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

Appendix: Data/Entity Migration Scenarios

Due to the updates in data and model provider options in 6.0 and 6.6, there are various scenarios for migrating your data and entities and best practices for each.

In-place Migrations

Generally, for in-place migrations, you do not need to delete the ThingworxStorage and ThingworxBackupStorage folders or import data/entities after installing ThingWorx. Wiki data is an exception, as it must be exported and then imported after upgrade.

NOTE: If DSE is the default persistence provider for this in-place migration scenario, blog and wiki data will not be migrated. Only model entities are migrated. Export blog and wiki data and then import after migrating.

	Migrating To ThingWorx					
Migrating From ThingWorx		7.2 on Neo4j	7.2 on Neo4j with DSE	7.2 on PostgreSQL	7.2 on PostgreSQL with DSE	7.2 on H2
	5.X on Neo4j	Export / Import	Export / Import + DSE setup	Export / Import	Export / Import + DSE Setup	Export / Import ⁵
	6.0, 6.5, 6.6, 7.0, 7.1 on Neo4j	In Place ^{2,3,4}	In Place ^{2,3,4} + DSE Setup	Export / Import	Export / Import + DSE Setup	Export / Import ⁵
	6.0, 6.5, 6.6, 7.0, 7.1 on Neo4j with DSE ¹	--	In Place ^{2,3,4}	--	Export / Import	--
	6.5, 6.6, 7.0, 7.1 on PostgreSQL with DSE ¹	--	--	--	In Place ^{2,3}	--
	6.5, 6.6, 7.0, 7.1 on PostgreSQL	--	--	In Place ^{2,3}	In Place ^{2,3} + DSE Setup	--

¹ Also referred to as ThingWorx Enterprise Edition.

²: Can upgrade in place, migration optional for parallel implementation if desired.

³: Extensions need to be reimported.

⁴: In-place migrations for Neo4j should be done only for service pack level upgrades, not major or minor releases.

⁵: Exported applications built on Neo4j architecture might experience performance issues on H2. Migrating to PostgreSQL-based versions for larger applications that have high data volumes should be investigated instead.