MASHZONE NEXTGEN

# ADMINISTRATION

VERSION 10.15

OCTOBER 2022

This document applies to MashZone NextGen Version 10.15 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

# Contents

Contents ..... I

1 Preface ..... 1

2 MashZone NextGen security ..... 2
2.1 Manage your MashZone NextGen profile ..... 3
2.1.1 Manage your locale and account information ..... 3
2.1.2 Change your password ..... 3
2.2 MashZone NextGen server ..... 4
2.3 MashZone NextGen repository ..... 4
2.4 Encrypt database password ..... 4
2.5 Authentication and Guest Access ..... 6
2.5.1 User Authentication ..... 6
2.5.2 Valid Credentials ..... 7
2.5.3 Sessions and Timeouts ..... 7
2.6 Default user account ..... 8
2.7 Authentication with single sign-on solutions ..... 8
2.7.1 Configuration for Agent-Based SSO Solutions ..... 9
2.7.2 Implementing a Custom SSO Filter ..... 11
2.7.3 SSO integration in My webMethods ..... 11
2.8 Anti-Clickjacking prevention when using iFrame ..... 12
2.8.1 MashZone NextGen HTTP header security filter ..... 12
2.8.1.1 Example ..... 14
2.8.2 MashZone NextGen Content Security Policy ..... 14
2.8.3 Add a trusted site to allow iFrame ..... 14
2.8.4 Add multiple trusted sites to allow iFrame ..... 15
2.8.5 Content-Security-Policy using wildcards ..... 15
2.9 Handle personal data in log files ..... 16
2.10 Create whitelists for URL calls ..... 16

3 Getting Started with the MashZone NextGen Server ..... 19
3.1 Additional MashZone NextGen System and Software Requirements ..... 20
3.1.1 Additional Recommendations for MashZone NextGen ..... 20
3.2 What is Installed with MashZone NextGen ..... 21
3.3 Start and Stop the MashZone NextGen Server ..... 21
3.3.1 Start the MashZone NextGen Server ..... 21
3.3.2 Stop the MashZone NextGen Server ..... 22
3.4 Startup Considerations ..... 22
3.5 Manage Licenses for MashZone NextGen and BigMemory Max ..... 23
3.6 Move the MashZone NextGen repository to a robust database solution ..... 24
3.6.1 Troubleshooting Connections to the MashZone NextGen Repository ..... 24
3.6.2 Move the MashZone NextGen repository to Microsoft SQL Server ..... 25
3.6.3 Move the MashZone NextGen repository to MySQL ..... 27
3.6.4 Move the MashZone NextGen repository to Oracle ..... 29
3.6.5 Move the MashZone NextGen repository to PostGres ..... 30

3.7 Integrate Your LDAP Directory with MashZone NextGen ..... 32
3.7.1 Defining LDAP Connection Configuration ..... 33
3.7.2 Defining the Authentication Scheme ..... 34
3.7.3 Defining the Authorization Scheme ..... 35
3.7.4 Enabling MashZone NextGen Application Queries for All LDAP Users or Groups for Permissions ..... 37
3.7.5 Start MashZone NextGen with an initial administrator user ..... 38
3.8 Use the default MashZone NextGen user repository ..... 39
3.8.1 Manage users ..... 39
3.8.1.1 Create users ..... 40
3.8.1.2 Edit, assign roles and other user management tasks ..... 40
3.8.2 Manage user groups ..... 41
3.8.3 Which user roles exist? ..... 42
3.9 Install or unistall MashZone NextGen as Windows services ..... 42

4 MashZone NextGen Server Configuration ..... 44
4.1 Memory Configuration for the MashZone NextGen Server ..... 44
4.1.1 Configuration When MashZone NextGen Uses Only Heap Memory ..... 44
4.1.2 Configuration When MashZone NextGen Uses Heap and Off-Heap Memory ..... 45
4.2 Configure feed processing time zone ..... 46
4.3 Support International Character Sets and Locales ..... 47
4.3.1 Set the Repository Character Encoding ..... 48
4.3.2 Set the Repository Timezone or Offset ..... 48
4.3.3 Date, Time and Numeric Display Options ..... 49
4.3.4 Message Log and Default Locales ..... 49
4.4 Configure the MashZone NextGen server with custom ports ..... 50
4.4.1 Change MashZone NextGen Server Ports ..... 50
4.4.2 Change MashZone NextGen Repository Ports ..... 50
4.4.3 Tomcat Application Server Port ..... 51
4.5 Configure the MashZone NextGen server to work with a proxy server ..... 51
4.6 Embedding MashZone NextGen in external system environments ..... 52
4.6.1 Configure MashZone NextGen server to work with iFrame ..... 52
4.6.2 Post data ..... 53
4.6.3 URL selection ..... 54
4.7 Define a Proxy Server White list for MashZone NextGen ..... 54
4.7.1 Using Regular Expressions in a White list ..... 54
4.7.2 Specifying Literal Dot Separators ..... 55
4.7.3 Specifying Domains ..... 55
4.7.4 Specifying Host Names ..... 56
4.8 Configure MashZone NextGen for TLS and Digital Certificates ..... 56
4.8.1 The Certificate Store and Certificates ..... 57
4.8.2 Key Certificate Pairs ..... 57
4.8.3 Trusted Peer Certificates ..... 57
4.8.4 The Certificate Store ..... 58
4.8.5 Configure Mutual TLS Between Users and MashZone NextGen ..... 58
4.8.6 One-Way TLS to MashZone NextGen ..... 58

4.8.7 One-Way TLS to Information Sources....59
4.8.8 Configure HTTPS and Certificate Stores in the Application Server....59
4.8.9 Update TLS Configuration for Java....60
4.9 MashZone NextGen Logging....61
4.9.1 Configure Logging for the MashZone NextGen Server....61
4.9.2 Audit logging for dashboards, data feeds, aliases, and permissions....63
4.10 BigMemory Max for Caching, Connections and In-Memory Stores....64
4.10.1 Caching for the MashZone NextGen Server....65
4.10.1.1 Artifact Caching....65
4.10.1.2 Response Caching....65
4.10.1.3 Distributed Caching for MashZone NextGen Clusters....66
4.10.1.4 Configure BigMemory Max Servers for MashZone NextGen Caching....66
4.10.2 Working with BigMemory Max Stores for RAQL....68
4.10.2.1 Declared In-Memory Stores....68
4.10.2.1.1 Declare a new In-Memory Store....69
4.10.2.1.2 Modify a Declared In-Memory Store....71
4.10.2.1.3 View Details for Declared In-Memory Stores....71
4.10.2.1.4 Define permissions for declared In-Memory Strores....72
4.10.3 Dynamic In-Memory Stores....72
4.10.3.1 Manage Dynamic BigMemory Max Stores for MashZone NextGen Analytics....73
4.10.3.2 Add an External Dynamic In-Memory Store Connection....74
4.10.3.3 Delete External Dynamic In-Memory Store Connections....75
4.10.3.4 Define permissions for external dynamic In-Memory Store Connections....75
4.11 Manage data sources and drivers....76
4.11.1 Add a data source....76
4.11.2 Edit, test or remove data sources....78
4.11.3 Share data sources....78
4.11.4 Add or manage JDBC drivers....79
4.11.5 Migrate JDBC connections....80
4.11.5.1 Migrate JDBC configuration of Presto to MashZone NextGen....80
4.11.5.2 Migrate JDBC connections of Presto to MashZone NextGen....80
4.11.5.3 Migrate JDBC configuration of MashZone NextGen 9.10....81
4.11.5.4 Migrate JDBC connections of MashZone legacy to MashZone NextGen....82
4.12 Manage geographical map ressources....82
4.12.1 Manage geoJSON files....82
4.12.2 Manage tile server configuration files....83
4.13 Tune memory/caching for MashZone NextGen....84
4.13.1 Tune MashZone Memory and Cache Configuration Manually....84
4.13.2 Update Cache Memory Settings....85
4.13.3 Update MashZone ThreadSize Properties....85

4.14 Disable automatic masking in CSV files ..... 85

5 MashZone NextGen Server Administration ..... 86
5.1 Manage files for MashZone NextGen dashboards and data feeds ..... 86
5.1.1 Add external resources as MashZone NextGen files ..... 86
5.1.2 Find MashZone NextGen files ..... 87
5.1.3 Update or delete MashZone NextGen files ..... 87
5.1.4 Share MashZone NextGen resource files ..... 88
5.2 Manage resource directories ..... 88
5.2.1 Create resource directory ..... 89
5.2.2 Change resource directory ..... 89
5.2.3 Delete resource directory ..... 89
5.2.4 Share resource directory ..... 90
5.3 Manage URL aliases ..... 90
5.3.1 Create URL alias ..... 91
5.3.2 Change URL alias ..... 91
5.3.3 Delete URL alias ..... 91
5.3.4 Share URL alias ..... 92
5.4 Deploying MashZone NextGen instances, clusters, or artifacts ..... 92
5.4.1 Deploying the core widgets ..... 93
5.4.2 Deploying artifacts and other metadata using the command line ..... 93
5.4.2.1 Export users, groups, and role assignments ..... 95
5.4.2.2 Export role assignments for users and groups ..... 96
5.4.2.3 Export dashboards ..... 97
5.4.2.4 Export data feeds ..... 98
5.4.2.5 Export aliases ..... 99
5.4.2.6 Import users, groups, and role assignments ..... 100
5.4.2.7 Import role assignements for users and groups ..... 101
5.4.2.8 Import dashboards ..... 102
5.4.2.9 Import data feeds ..... 103
5.4.2.10 Import aliases ..... 104
5.4.2.11 Deploying multiple MashZone NextGen servers in one host ..... 104
5.4.3 Deploying artifacts using the Admin Console ..... 105
5.4.3.1 Export dashboards, data feeds, and aliases ..... 105
5.4.3.2 Import dashboards, data feeds, and aliases ..... 106
5.5 Clustering MashZone NextGen Servers ..... 107
5.5.1 Setting Up a New Cluster ..... 107
5.5.2 Adding New Members to an Existing Cluster ..... 109
5.6 Sharing the MashZone NextGen Repository in Clustered Environments ..... 110
5.6.1 Create and Share a New MashZone NextGen Repository ..... 110
5.6.2 Share an Existing MashZone NextGen Repository ..... 110
5.7 Setting Up an External MashZone NextGen Configuration Folder ..... 111
5.7.1 MashZone NextGen File-Based Configuration and Extensions ..... 112
5.7.2 MashZone NextGen Configuration Files That Can Be External ..... 113
5.7.3 MashZone NextGen Configuration Files That Must Be Internal ..... 114
5.7.4 MashZone NextGen Extensions ..... 115

5.8 MashZone NextGen dashboards in a clustered scenario ..... 115
5.8.1 Preliminary ..... 116
5.8.2 Configuration ..... 116
5.8.3 MashZone NextGen nodes ..... 116
5.8.3.1 Customizing dashboards ..... 116
5.8.4 Custom styles ..... 117
5.8.5 Custom widgets ..... 117
5.8.6 Using JDBC drivers ..... 117
5.8.7 Local file resources ..... 118
5.9 Customize application and dashboard styles ..... 118
5.9.1 Customize dashboard and widget styles ..... 118
5.9.2 Customize the application style ..... 119
5.9.3 Customize the MashZone NextGen welcome page ..... 120
5.9.4 Download styles ..... 121
5.9.5 Upload styles ..... 121
5.9.6 Compile styles ..... 122
5.9.7 Delete styles ..... 123
5.9.8 Switch the style editing mode ..... 124
5.9.9 Adjust the font size to different devices ..... 124
5.9.10 Custom widgets ..... 126
5.9.11 Style file structure ..... 127
5.10 Empty MashZone NextGen caches ..... 129

6 Event Service Configuration and Administration ..... 130
6.1 Manage Apama Instances ..... 130
6.1.1 Create Apama Instances ..... 130
6.1.2 Edit Apama Instances ..... 131
6.1.3 Delete Apama Instances ..... 131
6.2 Manage Apama Event Targets ..... 132
6.2.1 Create Apama Event Targets ..... 132
6.2.2 Edit Apama Event Targets ..... 133
6.2.3 Delete Apama Event Targets ..... 134
6.2.4 Share Apama Event Target ..... 134

7 Process Performance Manager Integration ..... 136
7.1 Manage PPM Connections ..... 136
7.2 Create PPM Connections ..... 136
7.3 Edit PPM Connections ..... 138
7.4 Delete PPM Connections ..... 139
7.5 Share PPM connections ..... 139

8 webMethods Business Console Integration ..... 141
8.1 Example ..... 141
8.2 Authentication ..... 142
8.3 Example URL ..... 142
8.4 Configuration ..... 142
8.5 Outbound API ..... 143
8.6 Inbound API ..... 143

9 MashZone NextGen Repositories ..... 144
9.1 Maintenance Suggestions ..... 144
9.2 Tuning the MashZone NextGen Repository Connection Pool ..... 145
9.2.1 Connection Pool Size Properties ..... 145
9.2.2 Idle Pool Connection Properties ..... 145
9.3 Synchronize the MashZone NextGen Repository and MashZone NextGen Server Time Zones ..... 146

10 Additional Information and Support ..... 147
10.1 Samples, Help and Other Documentation ..... 147
10.2 Version and License Information ..... 147

11 Legal information ..... 148
11.1 Documentation scope ..... 148
11.2 Support ..... 148

# 1 Preface

The MashZone NextGen Administration Guide includes information for administrators to configure and manage MashZone NextGen.

# 2 MashZone NextGen security

MashZone NextGen enables you to control user interactions including registering or creating dashboards and data feeds. You can also secure access for all users to work with these artifacts, based on policies that you define.

- **Change password**: For security reason, we strongly recommend that the MashZone NextGen administrator change the standard MashZone NextGen password after installation.
- **Change password of target data sources**: For security reason, we strongly recommend that you change the key used to encrypt or decrypt passwords of target data sources (for example, source operators, URL aliases, JDBC configurations). The key is included in the **authTokenKey** file located in <MashZone NextGen installation>/webapps/mashzone/WEB-INF/classes/. You can change the key by using the **padmin generateKey -a AES -f authTokenKey** command that creates a new **authTokenKey** file. First, you must create a backup of the existing **authTokenKey** file and then copy the new file to that folder. The file should be changed only with an empty repository, because already encrypted passwords can no longer be decrypted. The same applies to exported content. The system where the content is imported must use the same key to be able to decrypt the passwords.
- **User Authentication**: based on the protocols shown above. You can also allow anonymous access if needed. See Authentication and Guest Access (page 6) for details.
- Incorporate password policies and expiring passwords.

Consider the following security-relevant aspects :

- Always keep your operating system, installed widgets, and applications updated. Run necessary security updates on a regular basis, in particular for your web browser and installed plug-ins.
- Always keep your MashZone NextGen installation updated. Regularly check if new fixes are available for your installation and install them.
- To prevent unauthorized access to your system, only a limited number of users should be granted direct system access (for example, remote RDP access or directly using a management console).
- Limit network access by operating the server widgets behind a firewall. Only necessary services should be open in the firewall (for example, database services).
- Hide network ports used solely for internal communication between server widgets.
- Set up secure communication between the client and server using HTTPS. For details, see Configure HTTPS and Certificate Stores in the Application Server (page 59).

- Install the latest security updates of your operating system, browsers, and plug-ins, for example, Adobe Flash.

## 2.1 Manage your MashZone NextGen profile

Your user profile shows basic information about your account in MashZone NextGen and allows you to:

- Manage your locale and account information (page 3), if permitted
- Change your password (page 3), if permitted

### 2.1.1 Manage your locale and account information

In most cases, you cannot update any other account information because this comes from account information for your entire organization. In development or test environments where your account information is stored in the default MashZone NextGen Repository, you can save changes to this information.

**Procedure**

1. Click the  user icon in the program bar.
2. Click **My Profile**.
3. Make your settings.
4. Click **Save changes**.

Your settings are applied.

### 2.1.2 Change your password

In most cases, you **cannot** update your password in MashZone NextGen because this comes from account information for your entire organization. In development or test environments where your account information is stored in the default MashZone NextGen Repository, however, you can reset your password from your profile.

**Procedure**

1. Click the  user icon in the program bar.
2. Click **My Profile**.
3. Click **My Password**.
4. Enter your new password and confirm this.

5. Click **Update Password**.

Your password is updated.

## 2.2 MashZone NextGen server

**Security**: this includes both authentication and authorization for users when dashboards and data feeds are viewed or run. The MashZone NextGen server also handles authentication with dashboards and data feeds information sources when they are run.

The MashZone NextGen server is integrated with your user repository (page 4) or identity server for user authentication. This can be basic authentication, secure connections and certificates, or a single sign-on solution.

You define authorization policies for MashZone NextGen resources determining who can view or run dashboards and data feeds. Generally, users must be authenticated, but you can also define unlimited access, allowing 'guest' users without authentication to work with apps that are published to web sites, wikis or other environments.

## 2.3 MashZone NextGen repository

The MashZone NextGen repository contains information on users and groups, authorization policies, server configuration, and much more.

**User Data**: for authentication and determining authorization.

Typically user data comes from your organization's LDAP directory which you integrate with MashZone NextGen. This may also use a single sign-on solution and an identity manager. However, MashZone NextGen also has a built-in user repository which you may use. User or group meta-data from LDAP allows MashZone NextGen to relate authorization policies with users.

## 2.4 Encrypt database password

By default, the database password is specified in plain text. For security reasons, we recommend encrypting the database connection password.

The password is stored in the **context.xml** file located in the **<MashZone NextGen installation\apache-tomcat\conf** directory in the following form.

**Example**

```
<Resource name="MashzoneNextGenRepository"
```

```
auth="Container" type="javax.sql.DataSource" maxTotal="200" maxIdle="30"
maxWaitMillis="10000"
validationQuery="values(1)" username="app"
password="app"
driverClassName="org.apache.derby.jdbc.EmbeddedDriver"
url="jdbc:derby:${catalina.base}/bin/mashzonenextgenrepository"/
```

### Procedure

1. Add the **factory** attribute with the **com.softwareag.mashzoneng.tomcat.MZNGDataSourceFactory** class reference to the **Resource** definition in the **context.xml** file.

   You must specify an encrypted password for all **Resource** definitions for which you have defined a **factory** attribute with the **MZNGDataSourceFactory** class reference.
2. Create encrypted password using the **pAdmin** tool.
   a. Open a command line in the **<MashZone NextGen installation\prestocli\bin** folder.
   b. Run the following command.

      pAdmin generateEncryptedPassword -u Administrator -w manage -l http://localhost:8080/mashzone -e manage -f secure.txt

      The command decrypts the **manage** passwort specified after the **-e** parameter and stores it in the **secure.txt** file.
   c. Open the **secure.txt** file and copy the encrypted password to clipboard.
   d. Replace the value of the **password** attribute with the encrypted password in the **context.xml** file.
   e. Save your settings.

You have specified an encrypted password for the database connection.

### Example

The definition of the data source can be as follows.

```
<Resource name="MashzoneNextGenRepository"
auth="Container" type="javax.sql.DataSource" maxTotal="200" maxIdle="30"
factory="com.softwareag.mashzoneng.tomcat.MZNGDataSourceFactory"
maxWaitMillis="10000"
validationQuery="values(1)" username="app"
password="UNunEfwrajxJ5kPHru8Swg=="
driverClassName="org.apache.derby.jdbc.EmbeddedDriver"
url="jdbc:derby:${catalina.base}/bin/mashzonenextgenrepository"/>
```

Below are the parameters of the **generateEncryptedPassword** call of the **padmin** command.

padmin generateEncryptedPassword -u user -e encryptPassword [-v]

[-f outputFile] -w password [-l prestoUrl]

-l,--prestoUrl <prestoUrl> (optional) server URL - default http://localhost:8080/mashzone

-e,--encryptPassword <encryptPassword> Specify cleartext password to encrypt.

-f,--outputFile <outputFile> (optional) output file

-u,--user <user> use given user

-v,--verbose verbose output

-w,--password <password> use given password

**Warning**

Note that each time you modify the **authTokenKey** file with the **pAdmin generateKey** command (see the **MashZone NextGen security** (page 2) chapter), you must create a new encrypted password and enter it in the **context.xml** file. Otherwise, MashZone NextGen would not start correctly if you have enabled the factory attribute in the context.xml file.

## 2.5 Authentication and Guest Access

MashZone NextGen accepts requests from **both** unauthenticated (guests) and authenticated users.

Authentication is required:

- To use any feature in any MashZone NextGen Add-On that accesses the MashZone NextGen Server, unless that Add-On also supports guest access.

Requests are rejected with an authentication error when they do not provide one of:

- A valid MashZone NextGen session cookie. Sessions that have timed out are rejected with an appropriate error. See Sessions and Timeouts (page 7) for more information.
- Valid credentials. See Valid Credentials (page 7) for more information.
- Guest access header or parameter information.

### 2.5.1 User Authentication

MashZone NextGen is initially installed with a set of Default User Accounts (page 8) that you can use to get started. You configure MashZone NextGen to work with your LDAP Directory or you can continue to use the Default User Repository and simply add users and user groups to MashZone NextGen. See Use the Default MashZone NextGen User Repository (page 39), Manage Users (page 39) and Manage User Groups (page 41) for more information.

Authentication to verify user identities is performed against LDAP or the default User Repository and uses one of these protocols:

- Basic authentication with username and password
- This is the default authentication mechanism. No additional configuration is needed.
- TLS and User Certificates
- A configurable Single Sign-On solution
- See Authentication with single sign-on solutions (page 8) for configuration instructions.

Permission to work with MashZone NextGen artifacts can also be granted to guests (unauthenticated users), if needed.

### 2.5.2 Valid Credentials

When authentication is required, requests must have a valid MashZone NextGen session for an existing authenticated user or must supply either user credentials or digital certificate for authentication or an SSO token or ticket for a user that has been authenticated by the SSO solution. MashZone NextGen uses certificates, tokens or tickets to obtain the user's identity.

MashZone NextGen supports the following mechanisms to obtain user credentials or user IDs:

- Basic authentication using username and passwords. This is authenticated against the MashZone NextGen User Repository which may be a database or your LDAP Directory. See Use the Default MashZone NextGen User Repository (page 39) for more information.
- TLS and Certificate authentication where the user identifier in certificate information is configurable. This is authenticated against the MashZone NextGen User Repository which may be a database or your LDAP Directory, **unless** Dynamic User Support is enabled. See Use the Default MashZone NextGen User Repository (page 39) for more information.
- Single sign-on (SSO) solutions which are configurable. With SSO enabled, MashZone NextGen delegates authentication to the SSO solution. Typically, configuration identifies an SSO token, ticket or cookie that MashZone NextGen uses to verify authentication with the SSO solution and to obtain the user ID. See Authentication with Single Sign-On Solutions (page 8) for more information.

If an authenticated request has no MashZone NextGen session, MashZone NextGen starts a new session and generates a MashZone NextGen session cookie. See Sessions and Timeouts (page 7) for more information.

### 2.5.3 Sessions and Timeouts

MashZone NextGen is based on the standard J2EE session mechanism supported by your application server. MashZone NextGen maintains a separate HTTP session for each

authenticated user that has a unique session cookie. Each request with a valid MashZone NextGen session cookie extends the timeout limit for that user session.

SSO solutions maintain their own sessions and may use their own session cookies. SSO session cookies can be used to authenticate users in MashZone NextGen. SSO sessions and MashZone NextGen session are separate.

The default session timeout for MashZone NextGen is 30 minutes, defined in the web-apps-home/mashzone/WEB-INF/web.xml configuration file. In general, HTTP session timeouts can be configured in **web.xml**, unless the application server provides other configuration mechanisms. Please see your application server documentation for additional information on session configuration.

## 2.6 Default user account

MashZone NextGen has one user account that you can use 'out-of-the-box' to access MashZone NextGen dashboards and data feeds.

| Username | Password | Built-in Group / Permissions | Description |
|---|---|---|---|
| Administrator | manage | Administrator | A MashZone NextGen administrator. |

If you configure MashZone NextGen to use your LDAP Directory as the MashZone NextGen User Repository, the default user account is automatically disabled. If you use the Default User Repository, you can delete this user account in the Admin Console.

**Important:** You must ensure that at least one user account has MashZone NextGen administrator role.

## 2.7 Authentication with single sign-on solutions

With a single sign-on (SSO) solution, MashZone NextGen delegates authentication to the SSO layer. MashZone NextGen has the following pre-configured options to integrate with SSO solutions:

- Agent-based SSO solutions, such as Netegrity SiteMinder. See Configuration for Agent-Based SSO Solutions (page 9) for instructions.
- MashZone NextGen provides the integration under My webMethods in a SSO scenario by SAML (Security Assertion Markup Language).
- See SSO integration in My webMethods (page 11) for details.

### 2.7.1 Configuration for Agent-Based SSO Solutions

MashZone NextGen delegates authentication to the SSO layer, but expects user identity information from the SSO layer in the request in either an HTTP header or a parameter in the request URL. MashZone NextGen uses an extractor to find identity information in the header or parameter, and uses a transformer, to derive the user ID from the identity information. MashZone NextGen then uses the user ID to perform authorization and process the request.

To configure MashZone NextGen to work with an agent-based SSO layer, you configure the extractor and the transformer layers to work with your SSO solution and the identity information for your environment. MashZone NextGen provides a default extractor that looks for an HTTP header or parameter by name. MashZone NextGen also provides default transformers that handles cases where the identity information is just the user ID or can be found within the identity information using a regular expression.

You can also implement custom extraction or transformation layers to integrate MashZone NextGen with your SSO solution. See Implementing a Custom SSO Filter (page 11) for details.

**Procedure**

1. If needed, configure the MashZone NextGen User Repository. See Use the Default MashZone NextGen User Repository (page 39) for more information.

   In previous releases, MashZone NextGen only supported SSO solutions with LDAP as the MashZone NextGen User Repository. This restriction no longer applies.
2. Change the SSO filter in the applicationContext-security.xml configuration file for the MashZone NextGen Server:
   a. Open applicationContext-security.xml in any text or XML editor.

      This file is located in the web-apps-home/mashzone/WEB-INF/classes folder.
   b. Comment out the SSO filter bean (<bean id="ssoProcessingFilter">) for agent-based solutions (class="com.jackbe.jbp.sas.security.ui.sso.SSONullPreAutheticatedFilter").

      For example:

      ```
      <bean id="ssoProcessingFilter" > <property
      name="authenticationManager" ref="authenticationManager" />
      <property name="continueFilterChainOnUnsuccessfulAuthentication"
      value="true" /> ... </bean>
      ```

      Comment out the complete XML element with its children.
   c. Comment in the bean <bean id="ssoProcessingFilter
      class="com.jackbe.jbp.sas.security.ui.sso.SSOPreAuthenticatedFilter">.

      Comment in the complete XML element with its children.
3. In the agent-based SSO filter bean, configure the principalExtractor property:

   The default extractor uses a bean with the **HttpHeaderOrParamTokenExtractor** class.

<bean id="ssoProcessingFilter" > <property name="authenticationManager" ref="authenticationManager" /> <property name="continueFilterChainOnUnsuccessfulAuthentication" value="true" /> <property name="principalExtractor"> <bean > <property name="httpHeaderName" value="SM_USER"/> </bean> </property> ... </bean>

Change the value of the **httpHeaderName** property for this extractor bean to the name of the HTTP header or parameter that contains user identify information from your SSO solution.

If you have a custom extractor class, replace the default extractor bean with configuration for your custom class.

4. In the agent-based SSO filter bean, configure the principalTransformer property:

   The default transformer property uses a bean with the **RegexExtractionStringTransformation** class. This uses a regular expression to extract some portion of the value for the SSO header or parameter to get the final user ID that MashZone NextGen can use for authorization checks.

   ```
   <bean id="ssoProcessingFilter" > <property name="authenticationManager" ref="authenticationManager" /> <property name="continueFilterChainOnUnsuccessfulAuthentication" value="true" /> <property name="principalExtractor"> <bean > <property name="httpHeaderName" value="SM_USER"/> </bean> </property> <property name="principalTransformation"> <bean > <constructor-arg index="0" value="CN=(.*?),"/> </bean> </property> </bean>
   ```

   If the value of the SSO solution header or parameter contains more than just the user ID, for example a full DN from LDAP for a user, you can change the regular expression in the <constructor-arg/> parameter for the default bean to extract the user ID. The default regular expression extracts the CN portion of a user DN from an LDAP Directory.

   If the value of the SSO solution header or parameter is **just** the user ID, no further transformation is needed. Change the **principalTransformer** bean to do nothing using the **NoOpStringTransformation** bean:

   ```
   <bean id="ssoProcessingFilter" > <property name="authenticationManager" ref="authenticationManager" /> <property name="continueFilterChainOnUnsuccessfulAuthentication" value="true" /> <property name="principalExtractor"> <bean > <property name="httpHeaderName" value="SM_USER"/> </bean> </property> <property name="principalTransformation"> <bean /> </property> </bean>
   ```

   If you have a custom transformation class, replace the default transformer bean with configuration for your custom class.

5. Save this file and restart the MashZone NextGen Server. See Start and Stop the MashZone NextGen Server (page 21) for instructions.

## 2.7.2 Implementing a Custom SSO Filter

If the default extractor and transformer filters available in MashZone NextGen do not provide the functionality needed to allow MashZone NextGen to work with your SSO solution, you can create custom filters using the MashZone NextGen SSO Filter API.

**Procedure**

1. Add the following JARs and classes to your classpath:

   Classes in the web-apps-home/mashzone/WEB-INF/classes folder.

   The web-apps-home/mashzone/WEB-INF/lib/presto_common.jar file.

2. Implement one or both filters:

   To create a custom extractor, implement the **SSOTokenExtractor** interface, typically using the **AbstractSSOTokenExtractor** base class.

   To create a custom transformer, implement the **Transformation** interface.

3. Add these classes to the classpath. Copy either the compiled class file or a JAR containing the compiled class file to one of these folders, respectively:

   The external configuration folder, if any, for the MashZone NextGen Server. See Setting Up an External MashZone NextGen Configuration Folder (page 111) for more information.

   **Important:** Deploying additional resources, such as custom SSO filters, to an external configuration folder simplifies future deployments or MashZone NextGen Server clusters.

   web-apps-home/mashzone/WEB-INF/classes. This is the default location, but is not recommended as it complicates MashZone NextGen Server deployments.

   web-apps-home/mashzone/WEB-INF/lib. This is the default location, but is not recommended as it complicates MashZone NextGen Server deployments.

## 2.7.3 SSO integration in My webMethods

You can integrate MashZone NextGen under My webMethods in an SSO scenario by SAML (Security Assertion Markup Language ).

MashZone NextGen can accept SAML tokens for authentication in a SSO environment. Specifically, My webMethods can act as an Identity Provider (IdP).

MashZone NextGen verifies the signature used to sign the SAML assertion is trusted by looking the comparing the signature to the **platform_truststore.jks** file. This file is a Java Keystore file, and can be managed using the Java "keytool" command. If the certificate used to sign the SAML assertion is not present in the **platform_truststore.jks** file, the assertion is rejected. The **platform_truststore.jks** file is configurable in

SAG_HOME/MashZoneNG/apache-tomcat/webapps/mashzone/WEB-INF/classes/presto.config.

Information on the Java "keytool" command can be found in the Java documentation: http://docs.oracle.com/javase/8/docs/technotes/tools/windows/keytool.html.

**Procedure**

1. Within the presto.config file, the saml.truststore.file parameter contains the full path to the file. The default configuration uses the SAG_HOME/common/conf/platform_truststore.jks file. By default, the file contains the certificate used to sign My webMethods SAML assertions. No further configuration is needed in the My webMethods SAML case.
2. Within the presto.config file, the saml.truststore.passwd parameter contains the keystore password. The default configuration uses the password for the SAG_HOME/common/conf/platform_truststore.jks file. The default password is manage.
3. To accept SAML assertions signed by a third party, the signing certificate must be either imported as a "trusted certificate" to the currently configured platform_truststore.jks file, or the presto.config file must be altered to point to a different keystore file, where this signing certificate is already imported as a "trusted certificate".

## 2.8 Anti-Clickjacking prevention when using iFrame

For security reason we recommend to configure your iFrame setting to protect your MashZone NextGen installation against clickjacking attacks.

Clickjacking is a vulnerability where an attacker creates a page that uses iFrame to render another page, then creates invisible controls on top of the rendered page that may be able to sniff user input.

General information on the clickjacking attack vector can be found on https://www.owasp.org/index.php/Clickjacking.

MashZone NextGen prevents clickjacking attacks by using the Content-Security-Policy that is supported by most web browsers. Details on how to use iFrame with MashZone NextGen can be found in Configure MashZone NextGen server to work with iFrame (page 52).

### 2.8.1 MashZone NextGen HTTP header security filter

MashZone NextGen provides a specific HTTP header security filter included in the **web.xml** file. By default, this filter always sends the X-Frame-Option: **SAMEORIGIN**, that can be configured to send **ALLOW-FROM** to any number of trusted websites. This HTTP response

header instructs the browser to refuse to render any content from MashZone NextGen in an iFrame, unless the iFrame is within MashZone NextGen itself.

**HttpHeaderSecurityFilter**

Following the commented configuration in the **web.xml** file.

```
<filter> <filter-name>HTTP Header Security Filter</filter-name>
<filter-class>
com.jackbe.jbp.sas.security.ui.http.HttpHeaderSecurityFilter
</filter-class> <!-- Init Param: antiClickJackingEnabled Should the anti
click-jacking header (X-Frame-Options) be set on the response. Valid options:
true or false When true, X-Frame-Options will always contain "SAMEORIGIN".
This instructs browsers to disallow iframing of MzNG content outside of the
MzNG application itself. If false, X-Frame-Options will not be sent at all,
which completely disables clickjacking protection allows any site to iframe
MzNG) Note: X-Frame-Options is superseded by Content-Security-Policy.
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
--> <init-param> <param-name>antiClickJackingEnabled</param-name>
<param-value>true</param-value> </init-param> <!-- Init Param:
antiClickJackingUris List of comma separated Uris for sites allowed to iframe
content in MzNG. To allow external sites to iframe MzNG content, uncomment
this init param, and add the site uri to the list. Also configure the 'Content
Security Policy' filter below. If the request to MzNG contains a referer value
matching the scheme, host and port of one of the Uris in the list, the
X-Frame-Options header will send "ALLOW-FROM uri'. This allows the browser
to render the iframe. If there is no match (or the list is empty) X-Frame-Options
will send "SAMEORIGIN" and the browser will refuse to render the iframe Any
site added to this list should also be added to 'Content Security Policy' header.
<init-param> <param-name>antiClickJackingUris</param-name>
<param-value>http://some-server.com</param-value> </init-param> --> <!--
Init param: hstsEnabled Enable HTTP Strict Transport Security (HSTS) header
(Strict-Transport-Security) to be set on the response for secure requests -->
<init-param> <param-name>hstsEnabled</param-name>
<param-value>true</param-value> </init-param> <!-- Init Param:
hstsMaxAgeSeconds The max age value that should be used in the HSTS header.
Negative values will be treated as zero. If not specified, the default value
of 0 will be used. --><init-param><param-name>hstsMaxAgeSeconds</param-name>
<param-value>604800</param-value> </init-param> </filter> <filter-mapping>
<filter-name>HTTP Header Security Filter</filter-name>
<url-pattern>/*</url-pattern> </filter-mapping>
```

The **antiClickJackingUris** parameters can take a list of comma separated URIs. The parameter is commented out by default. Any request for a MashZone NextGen resource containing a "Referer" header field matching the scheme, host and port of a URI in the **antiClickJackingUris** parameter will result in a response containing the X-Frame-Options response header with the appropriate **ALLOW-FROM** value. If there is no match, then the X-Frame-Options will carry the **SAMEORIGIN** value.

#### 2.8.1.1 Example

The Web site http://website-a.com is configured as trusted, and therefore it is listed in the **antiClickJackingUris** parameter, and contains a page that uses iFrame to embed a MashZone NextGen dashboard. When a user visits this page on website-a.com, the browser will attempt to fetch the iFramed dashboard from MashZone NextGen. The request generated by the browser will carry the HTTP request header "Referer" containing the full URI to the page containing the iFrame. MashZone NextGen will match the "Referer" URI with the trusted URI from **antiClickJackingUris** parameter, and recognize that the Web site is trusted. As a result, the response will carry the HTTP response header "X-Frame-Options: ALLOW-FROM http://website-a.com ". The browser will then allow the iFrame to render.

### 2.8.2 MashZone NextGen Content Security Policy

Most modern browsers such as Microsoft Edge, Chrome, Firefox and Safari check for the newer Content-Security-Policy HTTP header instead of X-Frame-Options. Within the MashZone NextGen **web.xml** file is a second HTTP filter class that sends the HTTP Header **Content-Security-Policy**. This filter is configured by default to send the value **frame-ancestors 'self'** which is equivalent to **SAMEORIGIN** in that it instructs the browser to only allow iFrame if the iFrame is already in the originating Web site.

The Content-Security-Policy is not supported by Microsoft Internet Explorer.

**ContentSecurityPolicy**

```
<filter> <!-- Allows setting of HTTP header Content-Security-Policy
http://www.w3.org/TR/CSP2/ To prevent clickjacking attacks default is
"frame-ancestors 'self'" which disallows external iframing of MzNG content.
To allow additional websites to iframe MzNG content, add the site Uri after
'self'. For example: "frame-ancestors http://*.example.com/ 'self'"
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/
Content-Security-Policy/frame-ancestors --> <filter-name>Content Security
Policy</filter-name>
<filter-class>com.jackbe.jbp.sas.security.ui.http.ContentSecurityPolicyFi
lter</ filter-class> <init-param> <param-name>policy</param-name>
<param-value>frame-ancestors 'self'</param-value> </init-param> </filter>
<filter-mapping> <filter-name>Content Security Policy</filter-name>
<url-pattern>/*</url-pattern> </filter-mapping>
```

### 2.8.3 Add a trusted site to allow iFrame

The default settings do not allow external sites to iframe internal MashZone NextGen assets such as dashboards, apps, etc. Specifically, "X-Frame-Options: SAMEORIGIN" and "Content-Security-Policy: frame-ancestors 'self'" are set, which instructs the browser to

disallow rendering MashZone NextGen content in any external iFrame. Via configuration and re-start, we can relax this restriction.

**Procedure**

1. Open the **applicationContext-security-filters.xml** file in a text editor. The file is located in <MashZone NextGen installation>/apache-tomcat/webapps/[presto|mashzone]/WEB-INF/classes.
2. Find the **<filter>** entry of the HTTP Header Security Filter and uncomment the **antiClickJackingUris** parameter.
3. Replace the sample URI ' http://some-server' with the URI of the Web site allowed to iframe MashZone NextGen content.
4. Find the **<filter>** entry for Content-Security-Policy. Insert the URI of the Web site allowed to iframe MashZone NextGen content into the **policy** parameter, between **frame-ancestors** and **'self'**.

   Example: **<init-param> <param-name>policy</param-name> <param-value>frame-ancestors http://*.eur.ad.sag:* 'self'</param-value> </init-param>**

## 2.8.4 Add multiple trusted sites to allow iFrame

To allow more than one Web site, perform the steps as shown in **Adding a trusted site to allow iFrame**.

**Procedure**

1. In the **HTTP Header Security** filter, add a comma separated list of URIs as the **antiClickJackingUris** value:

   ```
   <init-param> <param-name>antiClickJackingUris</param-name>
   <param-value>http://website-a.com, http://website-b.com:9999
   </param-value> </init-param>
   ```
2. In the **Content-Security-Policy** filter, add the URI to the policy parameter value, separated by a space:

   ```
   <init-param> <param-name>policy</param-name>
   <param-value>frame-ancestors http://website-a.com http://website-b.com
   'self' </param-value> </init-param>
   ```

## 2.8.5 Content-Security-Policy using wildcards

The Content-Security-Policy allows wildcards to be used in the policy. For example, to allow any Web site on any port hosted in the "eur.ad.sag" domain, you can specify:

```
<init-param> <param-name>policy</param-name> <param-value>frame-ancestors
http://*.eur.ad.sag:* 'self' </param-value> </init-param>
```

## 2.9 Handle personal data in log files

For some actions, MashZone NextGen tracks the user ID, IP address, email ID and full name of the executor. This data is used to analyze and fix potential problems that occur during system operation. The data is also stored after deletion of a user account. This ensures that no past user data is lost and all user data is accounted for in future audits.

You can remove this personal data from your MashZone NextGen system for General Data Protection Regulation (GDPR) compliance.

To remove the relevant personal user data, you must delete the corresponding log files from the MashZone NextGen system.

The relevant log files are stored in the following directories:

- IP address and user ID

  <MashZone NextGen Installation>\apache-tomcat\logs\...

  For example, localhost_access_log.2018-05-24
- Username

  MashZone NextGen Installation>\apache-tomcat\logs\MashZone-AuditLog.log
- IP Address

  MashZone NextGen Installation>\apache-tomcat\logs\wrapper.log

**Warning**

If you delete the log files, all logged data is lost and cannot be restored.

**Procedure**

In Windows® Explorer, go to the directories mentioned above and delete all relevant log files.

The user data is deleted from the MashZone NextGen system.

## 2.10 Create whitelists for URL calls

You can control outgoing HTTP requests when calling external URLs using URL whitelists. Use whitelists to prevent attackers from abusing HTTP requests to collect sensitive data, such as server configuration details in cloud scenarios, or to redirect users to phishing sites.

You can define URL whitelists that contain individual URLs that a user can use to access external sources. MashZone NextGen distinguishes between server-side and client-side requests based on URLs. On server-side are several data source operators that allow the data request from external URLs, for example, CSV, Excel, JSON, XML, PPM, and ARIS table. On the client-side, for example, it is the **Image** widget and the actions you can specify for multiple widgets.

Server- and client whitelists are stored in the MashZone NextGen database repository. The size of each whitelist is limited to 4000 characters. The whitelist is a JSON array of regular expressions, see the example below. URLs that match at least one regular expression have passed the check.

### Example

The following JSON array represents a URL whitelist.

```
[
    "https://www.softwareag.com.*",
    "https://github.com/w3c/csvw.*",
    "hhttps://www.w3schools.com/xml.*"
]
```

To create a URL whitelist, you can use an appropriate text editor. With the **padmin** command line tool, you can provide the created whitelist to MashZone NextGen. The padmin tool is located in the following directory:

<MashZone NextGen installation>/prestocli/bin/padmin.bat

Use the following commands to import your URL whitelist into MashZone NextGen:

- On server-side

  padmin importBackendUrlWhitelist -f backend_whitelist.json -u Administrator -w manage -t default

- On client-side

  padmin importFrontendUrlWhitelist -f frontend_whitelist.json -u Administrator -w manage -t default

You can also export your URL whitelist using the following commands. The server returns a JSON array of regular expressions, as shown in the above example, written to the specified file in the file system.

- On server-side

  padmin exportBackendUrlWhitelist -f backend_whitelist.json -u Administrator -w manage -t default

- On client-side

  padmin exportFrontendUrlWhitelist -f frontend_whitelist.json -u Administrator -w manage -t default

# 3 Getting Started with the MashZone NextGen Server

You install MashZone NextGen using the Software AG Installer. See the Installing Software AG Products guide for instructions.

The post-installation tasks you must complete to allow users to start working with MashZone NextGen include.

**Procedure**

1. Start the MashZone NextGen. See Start and Stop the MashZone NextGen Server (page 21) for instructions.
2. Login to MashZone NextGen:
   a. Open MashZone NextGen in a browser at http://localhost:8080/mashzone.

      If you used a different port number when you installed MashZone NextGen or the MashZone NextGen Server is running on a different host, change the domain and port number appropriately.
   b. Use the credentials for the default administrator account:

      User name = **Administrator**

      Password = **manage**
3. If you are using the default MashZone NextGen User Repository rather than an LDAP Directory to manage users and groups for MashZone NextGen, it is a good practice to change the password for this default administrator account.
   a. Open your profile from the MashZone NextGen Hub menu bar and click My Password.
   b. Enter your new password and confirm this.
   c. Then click Change Password.

   If you will use your LDAP Directory as the MashZone NextGen User Repository, this default account is disabled once LDAP configuration is complete.
4. Set up a robust database to use as the MashZone NextGen Repository.

   MashZone NextGen is installed with Derby as an embedded database hosting the MashZone NextGen Repository for trial purposes only. The default Derby database should **not** be used for serious development environments or for staging or production.

   See Move the MashZone NextGen repository to a robust database solution (page 24) for instructions.
5. If you want MashZone NextGen to use your LDAP Directory as the repository for user and group information, you must update configuration. See Integrate Your LDAP Directory with MashZone NextGen (page 32) for instructions.

6. If you have also installed BigMemory Max and received your BigMemory Max license, add this license to MashZone NextGen and configure MashZone NextGen to work with BigMemory Max. See Manage Licenses for MashZone NextGen and BigMemory Max (page 23) and Configure BigMemory Max Servers for MashZone NextGen Caching and In-Memory Stores (page 66) for instructions.

## 3.1 Additional MashZone NextGen System and Software Requirements

- For basic requirements to install MashZone NextGen, see the System Requirements for Software AG Products guide.

### 3.1.1 Additional Recommendations for MashZone NextGen

In addition to the basic recommendations in the System Requirements for Software AG Products guide, you should also consider the following recommendations for MashZone NextGen:

- A robust, compatible database to host the MashZone NextGen Repository is required.

  **Important:** The MashZone NextGen repository is initially installed in a Derby database suitable only for trial purposes. For proof-of-concept, development or production uses, move the repositories to a robust and compatible solution. See System Requirements for Software AG Products for more information.
- Architecture and memory requirements or recommendations include:

  64-bit architecture

  2G minimum of memory if only small to medium datasets are involved

  4G minimum of memory if large datasets are involved

  **Important:** Actual memory and disk space requirements vary widely based on load, throughput, performance and other requirements unique to your environment. Please contact your Software AG representative for more information.
- To ensure a secure connection between MashZone NextGen server and client it is recommended to operate your MashZone NextGen system via HTTPS as communication protocol. You can configure your application server accordingly after you have installed MashZone NextGen. See Configure HTTPS and Certficate Stores in the Application Server (page 59) for details.

## 3.2 What is Installed with MashZone NextGen

MashZone NextGen initially installs the applications **MashZone NextGen Server**.

MashZone NextGen also installs the following additional software:

- Apache's Tomcat Servlet Container, version 8.5.30
- Derby Database, version 10.13.1.1

  **Important:** The MashZone NextGen repository is initially installed in a Derby database suitable only for trial purposes. For proof-of-concept, development or production uses, move the repositories to a robust and compatible solution. For details, see the chapter Move the MashZone NextGen repository to a robust database solution (page 24).

## 3.3 Start and Stop the MashZone NextGen Server

Most MashZone NextGen widgets depend on the MashZone NextGen Server.

The MashZone NextGen Server depends on the MashZone NextGen Repository.

### 3.3.1 Start the MashZone NextGen Server

**Procedure**

1. If the MashZone NextGen Repository has been moved from the default Derby database and they are not already running, manually start these databases following the instructions for their host database.
2. Do one of the following to start the MashZone NextGen Server:
   a. For Windows systems, either:

      From the Start menu, select Software AG > Start Servers > Start MashZone NextGen version.

      Enter this command in a command window:

      **c:>MashZoneNG-install/apache-tomcat/bin/startup.bat**

      On Windows Server operating systems MashZone NextGen Server must be started as Administrator. To run MashZone NextGen Server as Administrator, right click on the **Start MashZone NextGen version** shortcut and select **Run as administrator**.

      Starting **startup.bat** from the file system using Windows Explorer does not work.

   b. For Linux, Mac OS X or UNIX systems, open a new terminal window and move to this folder:

      **% cd MashZoneNG-install/apache-tomcat/bin**

Then enter this command:

**% startup.sh**

3. Open the MashZone NextGen at **http://app-server:port/mashzone** and log in.

   You can now access all the MashZone NextGen tools: Feed Editor, Dashboard Editor, Dashboard Viewer and the Admin Console.

### 3.3.2 Stop the MashZone NextGen Server

**Procedure**

1. Do one of the following:

   For Windows systems, either:

   From the Start menu, select **Software AG > Stop Servers > Stop MashZone NextGen**.

   Enter this command in a command window:

   **c:>MashZoneNG-install/apache-tomcat/bin/shutdown.bat**

   For Linux, Mac OS X or UNIX systems, open a new terminal window and move to this folder:

   **% cd MashZoneNG-install/apache-tomcat/bin**

   Then enter this command:

   **% shutdown.sh**

2. If the MashZone NextGen Repository has been moved from the default Derby database, you can also choose to stop this database. See documentation for the host database for instructions.

## 3.4 Startup Considerations

When the MashZone NextGen Repository is hosted in a robust database solution, it must be started before the MashZone NextGen Server for a successful startup. With the default Derby database, the MashZone NextGen Repository runs as an **embedded** database that is automatically started with the MashZone NextGen Server.

In environments where you application server is started automatically with the host, this can create timing errors. You may need to stop and restart the MashZone NextGenMashZone NextGen Server after the MashZone NextGen Repository has been restarted.

If you host the MashZone NextGen Repository in a MySQL or Oracle database, you may also be able to have the database start automatically with the host.

## 3.5 Manage Licenses for MashZone NextGen and BigMemory Max

To use MashZone NextGen a license is required.

If you are using BigMemory Max features that require this, you also need to make your BigMemory Max license available to the MashZone NextGen Server and/or the Integrated MashZone Server. See BigMemory Max for Caching, Connections and MashZone NextGen Analytics (page 64) for features that require this additional license.

You can apply licenses when you install MashZone NextGen, or you can install and use MashZone NextGen without a license for a trial period of 30 days. If you purchase MashZone NextGen after installation, you must manually apply the MashZone NextGen license. If needed, you can also manually apply a BigMemory Max license.

When MashZone NextGen runs with a READ ONLY license, all tools to create and edit data feeds and dashboards are unavailable.

**Procedure**

1. Save the attached license file(s) from the email(s).
2. For MashZone NextGen licenses, copy the MashZoneNGLicense.xml file into the MashZoneNG-install/apache-tomcat/conf folder.

   If MashZone NextGen is deployed in a cluster, copy the license file to this folder in every cluster member.
3. If a BigMemory Max license is required:
   a. Copy the license file terracotta.key to the MashZoneNG-install/apache-tomcat/conf folder.

      If MashZone NextGen is deployed in a cluster, you must copy this file to every cluster member.
   b. Add the following Java system property to the MashZone NextGen server configuration file <MashZone NextGen installation>/apache-tomcat/conf/wrapper.conf:

      **wrapper.java.additional.<n+1>=-Dcom.tc.productkey.path=MashZoneNG-install/apache-tomcat/conf/terracotta.key**

      Where n is the number of last additional Java parameter.

      If MashZone NextGen is deployed in a cluster, you must update the server configuration files for every cluster member.
4. Restart the MashZone NextGen Server. See Start and Stop the MashZone NextGen Server (page 21) for instructions.

## 3.6 Move the MashZone NextGen repository to a robust database solution

The MashZone NextGen repository is initially installed in a Derby database suitable only for trial purposes. For proof-of-concept, development or production uses, move the repositories to a robust and compatible solution.

You can host the MashZone NextGen repository in any database that MashZone NextGen supports. See Additional MashZone NextGen System and Software Requirements (page 20) in System Requirements for details.

You can move the repository to one of the following databases:

- **Microsoft SQL Server**: see Move MashZone NextGen repository to Microsoft SQL Server (page 25) for instructions.
- **MySQL**: see Move the MashZone NextGen repository to MySQL (page 27) for instructions.
- **Oracle**: see Move the MashZone NextGen repository to Oracle (page 29) for instructions.
- **PostGres**: see Move the MashZone NextGen repository to PostGres (page 30) for instructions.

### 3.6.1 Troubleshooting Connections to the MashZone NextGen Repository

The most common problem when the MashZone NextGen Server server does not restart successfully after you move the MashZone NextGen Repository to a new database is that is cannot connect to the MashZone NextGen Repository. To verify that this is the problem:

- Open the MashZone NextGen Server log file **wrapper.log** in your web application server's log directory. For Tomcat, this is:

  apache-tomcat/logs/wrapper.log
- Check for a log entry for **Cannot create PoolableConnectionFactory** near the end of the file. This error indicates the MashZone NextGen Server could not successfully connect to the new database.

Common causes for this error include:

- The database hosting the MashZone NextGen Repository is not running.
- If this is true, start the MashZone NextGen Repository and verify that it is up. Then restart the MashZone NextGen Server and confirm that this starts successfully.
- There are one or more firewalls between the MashZone NextGen Repository and the MashZone NextGen Server that are not configured to allow this connection.

This can only happen when the database for the MashZone NextGen Repository is hosted on a different server than the MashZone NextGen Server.

- Update the firewall configuration to allow this connection. Then restart the MashZone NextGen Server and confirm that this starts successfully.
- The URL or other connection configuration that you entered in MashZone NextGen for the MashZone NextGen Repository is incorrect.
- To correct an error in this case, edit the resource properties for the MashZone NextGen Repository in the MashZoneNG-install/apache-tomcat/conf/context.xml file.
- Then restart the MashZone NextGen Server and confirm that this starts successfully.
- Port or connection configuration for the database is not set up properly to allow connections from the MashZone NextGen Server. See documentation for your database for more information.

### 3.6.2 Move the MashZone NextGen repository to Microsoft SQL Server

**Procedure**

1. If you are using your LDAP Directory as the MashZone NextGen User Repository, ensure that at least one user in your LDAP Directory has administrator privileges for MashZone NextGen before you move the MashZone NextGen Repository. For details, see the chapter Start MashZone NextGen with an initial administrator user (page 38).

   When the MashZone NextGen User Repository is your LDAP Directory, the default administrator account (**Administrator** user) is disabled.

2. If you are hosting the MashZone NextGen Repository or MashZone Repository in a new database, create the database following SQL Server documentation. Keep the following points in mind:

   Make sure this database is supported by MashZone NextGen. See Additional MashZone NextGen System and Software Requirements (page 20) for details.

   The jTDS driver and the original Microsoft driver are available for Microsoft SQL Server. You must make different settings depending on the driver type selected. For details see the following steps.

   If you want MashZone NextGen to support international characters in meta-data for artifacts, make sure the database uses the UTF-16 character encoding and case insensitive collation. See documentation for your database for specific instructions.

   It is a best practice to require passwords for every database account that can access the MashZone NextGen Repository.

3. Start the database that will become host to the MashZone NextGen Repository, if it is not already up.
4. Copy the JAR file for the JDBC driver for your database to the following folder for each MashZone NextGen Server that uses this MashZone NextGen Repository:

   MashZoneNG-install/apache-tomcat/lib
5. Replace the JAR for the MashZone NextGen Repository:
   a. Remove the web-apps-home/mashzone/WEB-INF/lib/jackbe-presto-rds-postgresql-derby.jar JAR file for each MashZone NextGen Server that uses this MashZone NextGenMashZone NextGen Repository. You can delete this JAR or simply move it to a folder that is not in the classpath for the application server that hosts MashZone NextGen.
   b. Copy this JAR file:

      MashZoneNG-install/prestorepository/jackbe-presto-rds-oracle-mysql-mssql.jar

      To the web-apps-home/mashzone/WEB-INF/lib folder.
6. Open the MashZoneNG-install/apache-tomcat/conf/context.xml configuration file in the text editor of your choice.
7. For the MashZone NextGen Repository, edit the <Resource> element with an ID of MashzoneNextGenRepository and:
   a. Update the JDBC driver, URL and credential properties:

      Example for jTDS driver

      ```
      <Resource name="MashzoneNextGenRepository" auth="Container"
      type="javax.sql.DataSource" maxTotal="200" maxIdle="30"
      maxWaitMillis="10000" username="app" password="app"
      driverClassName="net.sourceforge.jtds.jdbc.Driver"
      url="jdbc:jtds:sqlserver://host-name:port/database" />
      ```

      The JTA managed property **must** be **false**.

      Example for original Microsoft driver

      ```
      <Resource name="MashzoneNextGenRepository" auth="Container"
      type="javax.sql.DataSource" maxTotal="200" maxIdle="30"
      maxWaitMillis="10000" username="app" password="app"
      driverClassName="com.microsoft.sqlserver.jdbc.SQLServerDriver"
      url="jdbc:sqlserver://host-name:port;databaseName=your_database" />
      ```
   b. If needed, update optional properties. See the official Tomcat Datasource Properties for a complete list of optional properties and information on defaults.

      Some common properties you may need to set include:

      **validationQuery = select 1**

      Common tuning properties for connections pools. See Tuning the MashZone NextGen Repository Connection Pool (page 145).

8. Save your changes to this file.

   If the MashZone NextGen Server does not start up successfully, see Troubleshooting Connections to the MashZone NextGen Repository (page 24) for suggestions.

9. Restart the MashZone NextGen Server to apply these changes.

   If the MashZone NextGen Server does not start up successfully, see Troubleshooting Connections to the MashZone NextGen Repository (page 24) for suggestions.

### 3.6.3 Move the MashZone NextGen repository to MySQL

**Procedure**

1. If you are using your LDAP Directory as the MashZone NextGen User Repository, make sure that at least one user in your LDAP Directory has administrator privileges for MashZone NextGen before you move the MashZone NextGen Repository. For details, see the chapter Start MashZone NextGen with an initial administrator user (page 38).

   When the MashZone NextGen User Repository is your LDAP Directory, the default administrator account (**Administrator** user) is disabled.

2. If you are hosting the MashZone NextGen Repository in a new database, create the database following the official MySQL documentation. Keep the following points in mind:

   Make sure this database is supported by MashZone NextGen. See Additional MashZone NextGen System and Software Requirements (page 20) for details.

   If you want MashZone NextGen to support international characters in meta-data for artifacts, set the character encoding and collation to UTF-8 when you create the database. See documentation for your database for specific instructions.

   For medium or larger MySQL databases that will host the MashZone NextGen Repository, you should increase the maximum allowed packet size, which defaults to 1MB, for the database.

3. Start the database that will become host to the MashZone NextGen Repository, if it is not already up.

4. Replace the JAR for the MashZone NextGen Repository:

   a. Remove the MashZoneNG-install/apache-tomcat/mashzone/WEB-INF/lib/jackbe-presto-rds-postgresql-derby.jar JAR file for each MashZone NextGen Server that uses this MashZone NextGen Repository. You can delete this JAR or simply move it to a folder that is not in the classpath for the application server that hosts MashZone NextGenMashZone NextGen.

   b. Copy this JAR file:

MashZoneNG-install/prestorepository/jackbe-presto-rds-oracle-mysql-mssql.jar

To the web-apps-home/mashzone/WEB-INF/lib folder.

5. Copy the MySQL JDBC driver jar file to MashZoneNG-install/apache-tomcat/lib.
6. Open the MashZoneNG-install/apache-tomcat/conf/context.xml configuration file in the text editor of your choice.
7. For the MashZone NextGen Repository, edit the <Resource> element with an ID of MashzoneNextGenRepository and:
   a. Update the JDBC driver, URL and credential properties:

      ```
      name="MashzoneNextGenRepository" auth="Container"
      type="javax.sql.DataSource" maxTotal="200" maxIdle="30"
      maxWaitMillis="10000" username="app" password="app"
      driverClassName="com.mysql.jdbc.Driver"
      url="jdbc:mysql://host-name/databasename" />
      ```

      For MySQL databases, it is **recommended** including the database name in data source URLs. If this information is omitted, testing the data source fails and may also cause errors with access to stored procedures.

      The JTA managed property **must** be **false**.

   b. If needed, update optional properties. See the official Tomcat Datasource Properties for a complete list of optional properties and information on defaults.

      Some common properties you may need to set include:

      **validationQuery = select 1 from dual**

      Common tuning properties for connections pools. See Tuning the MashZone NextGen Repository Connection Pool (page 145).
8. Start the MashZone NextGen Server to apply these changes. This also starts the MashZone Server.

   If the MashZone NextGen Server does not start up successfully, see Troubleshooting Connections to the MashZone NextGen Repository (page 24) for suggestions.

**Note**

Depending on your local MySQL settings, MashZone NextGen may display an error message, such as "Error: Packet for query is too large (xxx > yyy)". In this case, a file (for example, the JDBC driver library) was uploaded to MashZone NextGen that exceeds the size of the **max_allowd_packet** parameter in the MySQL configuration. After setting an appropriate value in the **mysql.cnf** file, this error should no longer occur. For details, please refer the corresponding MySQL documentation.

### 3.6.4 Move the MashZone NextGen repository to Oracle

**Procedure**

1. If you are using your LDAP Directory as the MashZone NextGen User Repository, make sure that at least one user in your LDAP Directory has administrator privileges for MashZone NextGen before you move the MashZone NextGen Repository. For details, see the chapter Start MashZone NextGen with an initial administrator user (page 38).

   When the MashZone NextGen User Repository is your LDAP Directory, the default administrator account (Administrator user) is disabled.
2. If you are hosting the MashZone NextGen Repository in a new database, create the database following the official Oracle documentation.

   Make sure this database is supported by MashZone NextGen. See Additional MashZone NextGen System and Software Requirements (page 20) for details.

   If you want MashZone NextGen to support international characters in meta-data for artifacts, set the character encoding to AL32UTF8 when you create the database. See documentation for your database for specific instructions.

   It is a best practice to require passwords for every database account that can access the MashZone NextGen Repository.
3. Start the database that will become host to the MashZone NextGen Repository, if it is not already up.
4. Replace the JAR for the MashZone NextGen Repository:
   a. Remove the web-apps-home/mashzone/WEB-INF/lib/jackbe-presto-rds-postgresql-derby.jar JAR file for each MashZone NextGen Server that uses this MashZone NextGenMashZone NextGen Repository. You can delete this JAR or simply move it to a folder that is not in the classpath for the application server that hosts MashZone NextGen.
   b. Copy this JAR file:

      MashZoneNG-install/prestorepository/jackbe-presto-rds-oracle-mysql-mssql.jar

      To the web-apps-home/mashzone/WEB-INF/lib folder.
5. Copy the JAR file for the JDBC driver for your database to the following folder for each MashZone NextGen Server that uses this MashZone NextGen Repository:

   MashZoneNG-install/apache-tomcat/lib
6. Open the MashZoneNG-install/apache-tomcat/conf/context.xml configuration file in the text editor of your choice.

7. For the MashZone NextGen Repository, edit the <Resource> element with an ID of MashzoneNextGenRepository and:
   a. Update the JDBC driver, URL and credential properties:

      ```
      <Resource name="MashzoneNextGenRepository" auth="Container"
      type="javax.sql.DataSource" maxTotal="200" maxIdle="30"
      maxWaitMillis="10000" username="app" password="app"
      driverClassName="oracle.jdbc.OracleDriver"
      url="jdbc:oracle:drivertype:@host-name:port:dbname" />
      ```

      The JTA managed property **must** be **false**.
   b. If needed, update optional properties. See the official Tomcat Datasource Properties for a complete list of optional properties and information on defaults.

      Some common properties you may need to set include:

      **validationQuery = select 1 from dual**

      Common tuning properties for connections pools. See Tuning the MashZone NextGen Repository Connection Pool (page 145).
8. Save your changes to this file.

   If the MashZone NextGen Server does not start up successfully, see Troubleshooting Connections to the MashZone NextGen Repository (page 24) for suggestions.
9. Restart the MashZone NextGen Server to apply these changes.

   If the MashZone NextGen Server wedoes not start up successfully, see Troubleshooting Connections to the MashZone NextGen Repository (page 24) for suggestions.

## 3.6.5 Move the MashZone NextGen repository to PostGres

**Procedure**

1. If you are using your LDAP Directory as the MashZone NextGen User Repository, make sure that at least one user in your LDAP Directory has administrator privileges for MashZone NextGen before you move the MashZone NextGen Repository. For details, see the chapter Start MashZone NextGen with an initial administrator user (page 38).

   When the MashZone NextGen User Repository is your LDAP Directory, the default administrator account (**Administrator** user) is disabled.
2. If you are hosting the MashZone NextGen Repository or MashZone Repository in a new database, create the database following the official PostGreSQL documentation. Keep the following points in mind:

   Make sure this database is supported by MashZone NextGen. See Additional MashZone NextGen System and Software Requirements (page 20) for details.

If you want MashZone NextGen to support international characters in meta-data for artifacts, set the character encoding to UTF8 when you create the database. See documentation for your database for specific instructions.

It is a best practice to require passwords for every database account that can access the MashZone NextGen Repository.

When you initialize the PostGres database that will host the MashZone NextGen Repository, you may need to specifically set the locale used by the database to ensure case-insensitive sorting.

3. Start the database that will become host to the MashZone NextGen Repository, if it is not already up.
4. Copy the JAR file for the JDBC driver for your database to the following folder for each MashZone NextGen Server that uses this MashZone NextGen Repository: MashZoneNG-install/apache-tomcat/lib.
5. Open rdsApplicationContext.xml under <MashZone NextGen installation>/apache-tomcat/classes and add the following keys to:

   <property name="jdoProperties"> ... <map> ... <entry key="javax.jdo.mapping.Schema" value="public"/> <entry key="datanucleus.identifier.case" value="LowerCase"/> ... </map> </property>
6. If you are using PostgreSQL version 9.x please open postgresql.conf under PostgreSQL-install/9.x/data and un-comment the following property and make sure it is set to off: standard_conforming_strings = off.
7. Open the MashZoneNG-install/apache-tomcat/conf/context.xml configuration file in the text editor of your choice.
8. For the MashZone NextGen Repository, edit the <Resource> element with an ID of MashzoneNextGenRepository and:
   a. Update the JDBC driver, URL and credential properties:

      ```
      name="MashzoneNextGenRepository" auth="Container"
      type="javax.sql.DataSource" maxTotal="200" maxIdle="30"
      maxWaitMillis="10000" username="app" password="app"
      driverClassName="org.postgresql.Driver"
      url="jdbc:postgresql://host-name:port/databasename" />
      ```
   b. The JTA managed property **must** be **false**.
   c. If needed, update optional properties. See the official Tomcat Datasource Properties for a complete list of optional properties and information on defaults.

      Some common properties you may need to set include:

      **validationQuery = select 1**

      Common tuning properties for connections pools. See Tuning the MashZone NextGen Repository Connection Pool (page 145).

9. Save your changes to this file.

   If the MashZone NextGen Server does not start up successfully, see Troubleshooting Connections to the MashZone NextGen Repository (page 24) for suggestions.

10. Restart the MashZone NextGen Server to apply these changes. This also restarts the MashZone Server.

    If the MashZone NextGen Server wedoes not start up successfully, see Troubleshooting Connections to the MashZone NextGen Repository (page 24) for suggestions.

## 3.7 Integrate Your LDAP Directory with MashZone NextGen

In many cases, users and authentication information for an organization is defined in an existing LDAP Directory. You can configure MashZone NextGen to use your LDAP Directory as the source for user and group information.

See the System Requirements for Software AG Products guide for information on MashZone NextGen support for specific LDAP Directory solutions.

**Procedure**

1. If the MashZone NextGen Server is not yet started, start MashZone NextGen. See Start and Stop the MashZone NextGen Server (page 21) for instructions.
2. Change MashZone NextGen configuration to use LDAP as the authentication provider.
   a. Edit the userRepositoryApplicationContext.xml file in the MashZoneNG-config folder with any text editor.

      This folder may be in the default location or in an external location. See Setting Up an External MashZone NextGen Configuration Folder (page 111) for more information.

   b. Remove the comment markers around this statement: <import resource="/userRepositoryApplicationContext-ldap.xml">.

   c. Comment out this statement: <import resource="/userRepositoryApplicationContext-jdbc.xml"> property.

      You cannot use both default authentication and LDAP authentication.

   The configuration should look something like this:

   <beans> <!-- Choose between the interal JDBC repository and LDAP comment/uncomment these two import statements --> <import resource="/userRepositoryApplicationContext-ldap.xml"> <!-- <import resource="/userRepositoryApplicationContext-jdbc.xml"> --> ... </beans>

3. Change MashZone NextGen configuration for the user attribute provider.

a. If it is not already open, edit the userRepositoryApplicationContext.xml file in the MashZoneNG-config folder with any text editor.

   This folder may be in the default location or in an external location. See Setting Up an External MashZone NextGen Configuration Folder (page 111) for more information.

b. Find the userAttributeProvider bean:

```
<bean id="userAttributeProvider" > ...
```

c. Remove comment markers around the ldapAttributeProvider bean reference in the providers property list.

   The configuration should now look something like this:

```
<bean id="userAttributeProvider" > <property name="providers"> <list>
<ref bean="ldapAttributeProvider"/> <ref
bean="internalUserAttributeProvider"/> </list> </property> </bean>
```

d. Save your changes to this file.

   Do **not** restart the MashZone NextGen Server until all other LDAP configuration steps have been completed.

4. Define configuration in the Admin Console in MashZone NextGen Hub for:

   Connections to the LDAP Directory. See Defining LDAP Connection Configuration (page 33).

   Authentication mechanisms. See Defining the Authentication Scheme (page 34).

   Authorization mechanisms. See Defining the Authorization Scheme (page 35).

   All user and group queries used in MashZone NextGen applications. See Enabling MashZone NextGen Application Queries for All LDAP Users or Groups for Permissions (page 37).

5. Ensure that there is at least one user with administrative permissions in MashZone NextGen, otherwise the system will be locked. For details, see the chapter Start MashZone NextGen with an initial administrator user (page 38).

6. Restart the MashZone NextGen Server.

   MashZone NextGen now uses LDAP as the user repository. You can now login using the user account assigned in earlier steps as a MashZone NextGen administrator.

   To grant access to other users, add them to an appropriate built-in MashZone NextGen user group in LDAP.

### 3.7.1 Defining LDAP Connection Configuration

**Procedure**

1. Open the MashZone NextGen Admin Console.

2. Expand **MashZone NextGen Repositories** and click **User Repository - LDAP**.
3. Set these properties for your LDAP Directory:

   **LDAP URL** = the URL to your LDAP directory. For example:

   **ldap://localhost:389/dc=somecompany,dc=com**

   **Directory User Name** = the distinguished name for the user account to connect to this LDAP Directory. This **must** be a privileged account. For example:

   **uid=admin,ou=system**

   **Directory Password** = the password for the user to connect to this LDAP Directory.
4. Change any advanced options (see Defining the Authentication Scheme (page 34), Defining the Authorization Scheme (page 35) and Enabling MashZone NextGen Application Queries for All LDAP Users or Groups for Permissions (page 37)).
5. Save your changes.

### 3.7.2 Defining the Authentication Scheme

Authentication against LDAP determines if a distinguished name exists for a user. This searches for a user entry based on a specific username. Search-based authentication works, for example, if user names are users' email addresses.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand **MashZone NextGen Repositories** and click **User Repository - LDAP**.
3. Set these properties in the Authentication Properties section:

   **User Search Base** = the base context for a user search in authentication. This produces a list of all users which is filtered with a combination of the User Search Filter and User Search Subtree properties to authenticate a user. For example:

   **ou=users,ou=system**

   **User Search Filter** = the relative filter to apply to search for users during authentication. The variable **{0}** is replaced with the user's username from login.

   This filter is based from the context defined in User Search Base. For example:

   **email={0}**

   This attribute **must** be the same attribute used in the **User ID Attribute Name** property.

   **User Search Subtree** = set this option if the search should be recursive through all levels of the Directory under the search base. If you clear this option, search only checks direct children of the search base.

**Use LDAP VLV Control for Sorting and Paging** = this option is set by default to allow MashZone NextGen to use **virtual list views** (VLV) to paginate and sort LDAP search results.

Most LDAP directories support VLV, so in most cases you can leave this option set. If your LDAP directory logs errors for "unsupported search control", you can use this option to turn VLV off.

**User ID Attribute Name** = the LDAP attribute that contains the username that users login with. For example:

**email**

This value becomes the user ID for all further security contexts, unless the User ID Pattern property is also set.

**User ID Pattern** = a regular expression that is applied to user login names to extract the user ID for all further security contexts. This is only applied after authentication occurs.

### 3.7.3 Defining the Authorization Scheme

MashZone NextGen permissions are assigned to user groups or to individual users.

Your existing LDAP users and groups can be used in MashZone NextGen to define permissions for specific dashboards, data feeds, and aliases. For more information on authorization, see Authorization Policies and Permissions.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand **MashZone NextGen Repositories** and click **User Repository - LDAP**.
3. If user membership is defined in group entries in your LDAP directory, set these properties:

   Set the **Search Groups for User Membership** option.

   Enter the beginning context for user group searches in the **Group Search Base** property.

   This is combined with the User Group Search Filter to find LDAP groups to determine user membership in groups that may have MashZone NextGen permissions. For example:

   **ou=groups**

   Enter the filter to apply in group searches in the **User Group Search Filter** property.

   This is combined with Group Search Base to find LDAP groups to determine user membership in groups that may have MashZone NextGen permissions. The variable **{0}** is replaced with the user's username from login. For example:

   **uniquemember={0}**

Enter the LDAP attribute in group entries that identifies a group in the **Group Name Attribute** property.

This attribute contains the name of user groups that is used in MashZone NextGen permissions. The default value is the group common name:

**cn**

If you change this property, you **must** also update the **Group Name Pattern** property.

If group IDs in your LDAP Directory are not simple common names (see Group Name Attribute), enter a regular expression in **Group Name Pattern** to identify the built-in MashZone NextGen groups.

For example:

**cn=(PRESTO_.*?)**

MashZone NextGen expects specific names for the built-in groups that you add to your LDAP Directory. These values are defined in the common name of the group. This property allows MashZone NextGen to find the expected values for built-in groups, but use the full correct group names for the groups for your organization.

4. If user membership is defined solely in user entries, set these properties:

   Clear the **Search Groups for User Membership** option.

   Enter the name of the LDAP attribute in user entries that identies the groups that users belong to in the **User Membership Attribute** property.

   If group IDs in your LDAP Directory are not simple common names, enter a regular expression in **Group Name Pattern** to identify the built-in MashZone NextGen groups.

   For example:

   **cn=(PRESTO_.*?)**

   MashZone NextGen expects specific names for the built-in groups that you add to your LDAP Directory. These values are defined in the common name of the group. This property allows MashZone NextGen to find the expected values for built-in groups, but use the full correct group names for the groups for your organization.

With these properties set, for example:

```
Search Groups for User Membership = true Group Search Base = ou= groups,ou=system
User Group Search Filter=uniquemember={0} Group Name Attribute = cn
```

And a username of **jwalker**, MashZone NextGen would search all entries in **ou=groups** where **uniquemember=jwalker**. The names for any of these groups would be the common name (cn) for the group entry.

If these properties were set instead:

```
Search Groups for User Membership = false User Membership Attribute = memberOf
```

The list of groups would consist of all values in the **memberOf** attribute in the **jwalker** user entry.

This list of group names would be compared to the built-in MashZone NextGen groups and to groups with permissions for artifacts to determine the full set of permissions for **jwalker**.

### 3.7.4 Enabling MashZone NextGen Application Queries for All LDAP Users or Groups for Permissions

MashZone NextGen queries the User Repository for user groups and users to enable you and other users to assign permissions for MashZone NextGen resources. To enable these queries you set properties in the Admin Console.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand **MashZone NextGen Repositories** and click **User Repository - LDAP**.
3. To enable queries for all users, set these properties:

   **User Search Base** (in Authentication Properties) = the base context for a search for all users. This is used with the All Users Search Filter and Search Subtree For All Users properties to get a result. For example:

   **ou=People**

   This property is also used to search for users during authentication. Consider both uses before changing its value.

   **All Users Search Filter** (in MashZone NextGen Queries) = the search filter, combined with User Search Base that is used to find all user entries. For example:

   **objectclass=inetOrgPerson**

   Ensure that the **objectclass=inetOrgPerson** attribute is set on the LDAP server.
4. **To support wildcard searches and d**efine the sort order for results, you must also define these properties:

   Attributes Used in Wildcard Search (in MashZone NextGen Queries) = a list of LDAP attributes, separated by commas, to search in for wildcard searches. This defaults to:

   **cn,uid**

   **User Sort By Attribute** (in MashZone NextGen Queries) = the LDAP attribute that should be used to sort the results of wildcard searches. This defaults to:

   **cn**
5. You must also define these properties so that Admin Console can display minimal user information:

**User First Name Attribute** (in MashZone NextGen Queries) = the LDAP attribute that holds users' first names.

**User Last Name Attribute** (in MashZone NextGen Queries) = the LDAP attribute that holds users' last names.

**User Email Attribute** (in MashZone NextGen Queries) = the LDAP attribute that holds users' email addresses.

6. To enable queries for LDAP groups that can be used to assign MashZone NextGen permissions:

   **Group Search Base** (in Authorization Properties) = the beginning context, combined with Filter to Find All Groups for Roles to find all LDAP groups that can be used to assign MashZone NextGen permissions. For example:

   **ou=groups**

   This property is also used to search for MashZone NextGen permissions during authorization. Consider both uses before changing its value.

   **Filter to Find All Groups for Permissions** = the search filter, combined with Group Search Base that is used to find all LDAP groups that may be used to assign MashZone NextGen permissions. For example:

   **objectclass=groupOfUniqueNames**

Trouble shooting: If your LDAP user with role Presto_Administrator does not work, it might be helpful to stop MashZone NextGen first, deactivate and reactivate your LDAP connection in MashZone NextGen and then restart MashZone NextGen again.

## 3.7.5 Start MashZone NextGen with an initial administrator user

You can specify an initial user with administrator permissions to grant access to MashZone NextGen. This user can manage all role assignments of users and user groups in the MashZone NextGen Admin Console. In this way you can ensure that there is at least one user with administrative permissions in MashZone NextGen.

**Procedure**

1. Stop the MashZone NextGen server.
2. Open the **wrapper.conf** file in an appropriate text editor.
3. Specify the user ID of the initial administrator as follows.

   wrapper.java.additional.40=-Dusermanagement.initialadminaccount=**myadminuser**

<myadminuser> must be a valid user in LDAP. The user receives administrator permissions in MashZone NextGen.

4. Save your settings.
5. Restart the MashZone NextGen server.

MashZone NextGen is started with the specified user with administrator permissions.

**Tip**

You can add LDAP users to the MashZone NextGen user repository and assign user roles to them. For details on how to add users and groups to MashZone NextGen, and to assign the relevant roles, see chapter Use the default MashZone NextGen user repository (page 39).

## 3.8 Use the default MashZone NextGen user repository

MashZone NextGen authenticates users against a repository and retrieve authorization from the repository. This can be either a database or an LDAP Directory.

MashZone NextGen is installed with a default user repository within the MashZone NextGen Repository. You can use this initially as you explore MashZone NextGen or keep as the permanent source for user information, if desired. The default user repository also contains Default User Accounts (page 8) that you can use initially.

The MashZone NextGen repository is initially installed in a Derby database suitable only for trial purposes. For proof-of-concept, development or production uses, move the repositories to a robust and compatible solution.

No configuration is required to use the default user repository. If you use this default, you manage user and group information using functions in the Admin Console. See these topics for more information:

- Manage Users (page 39)
- Manage User Groups (page 41)

### 3.8.1 Manage users

If you are using the default MashZone NextGen User Repository, MashZone NextGen administrators can add users, assign them to groups to grant them permissions for various actions, and otherwise manage users in the Admin Console to. See Create users (page 40) and Edit, assign roles and other user management tasks (page 40) for instructions.

If you have configured MashZone NextGen to use your LDAP Directory as the User Repository (page 32), you mange users and groups in LDAP and assign permissions in MashZone

NextGen. For details, see Edit, assign roles, and other user management tasks (page 40) and Manage user groups (page 41).

### 3.8.1.1 Create users

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **Users and Groups** tab and click **Users**.
3. Click **Add new user** and complete the following information:

   **User Name** must be unique. This is the end-user's login name and primary key for the user record. User names cannot exceed 256 characters.

   **First Name** and **Last Name** are optional.

   **Email**

   **Password** cannot exceed 50 characters. Confirm the password.

   Valid characters for the User Name and Password fields are defined by the database you use for the default MashZone NextGen User Repository.
4. Set the **Active** option, if needed, and click **Add this User**.
5. Add another user or click **Cancel** to close this form.

   The new user is now active in MashZone NextGen and has permissions as an authenticated user. They can also access any artifact that grants permissions to the All Authenticated Users group.

   To give new users access to the MashZone NextGen to create artifacts or simply to work with artifacts from other users, you must add them to other groups, either manually or automatically.

**Tip**

Edit, assign roles and other user management tasks (page 40)

### 3.8.1.2 Edit, assign roles and other user management tasks

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **Users and Groups** tab and click **Users**.
3. Find the user in the list using Search or scrolling. Management options include.

4. Click **Reset password** and enter a new password.
5. Click **Edit** to change the user's name, login name or email.
6. Click **Change user status** to activate the user or click **Change user status** to deactivate.
7. Click **Manage User Groups** to add users to user groups or remove users from user groups.
   a. Enter part of a group name, if needed, and click **Search** to see a list of user groups and the user groups currently assigned to this user.
   b. Click a group from the left column to assign a user group to this user. Click a group from the right pane to remove a user group.
   c. Click **Save changes** to update groups for this user.
8. Click **Edit user roles** to grant or manage permissions for a user.
   a. Click **Search roles** to list all available roles. Or enter a role name and click **Search roles** to search a specific role.
   b. Click a role name in the left column to assign the role to the user. Click a role in the **Assigned Roles** box to remove a role.
   c. Click **Save changes** to update the roles assigned to the user.
9. Click **Delete user** to delete a user and confirm. Note that there must be at least one active user with **Administrator** role assigned. Therefore it is not possible to deactivate or delete the last user with **Administrator** role assigned.

**Tip**

Which user roles exist in MashZone NextGen? (page 42)

## 3.8.2 Manage user groups

If you are using the default MashZone NextGen User Repository, MashZone NextGen administrators add and manage user groups in the Admin Console. If you have configured MashZone NextGen to use your LDAP Directory as the User Repository, you define and manage user groups in your LDAP Directory.

User groups contain of a set of users. Groups can have one or more roles assigned.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **Users and Groups** tab and click **Groups**.

A list of groups displays. You can filter this list by entering part of a group name and clicking **Search**.

3. Click **Add new user group** and:
   a. Enter a unique name in the **Group** field.
   b. Click **Add this Group**.
   c. Add more groups or click **Cancel** to close this form.
4. Click **Manage roles** to grant or manage permissions for this user group.
   a. Click **Search roles** to list all available roles. Or enter a part of a role name and click **Search roles** to search a specific role.
   b. Click a role name in the left column to assign the role to the user. Click a role in the **Assigned Roles** box to remove a role.
   c. Click **Save changes** to update the roles assigned to the user.
5. To delete a user group, click **Delete** and confirm this.

**Tip**

Which user roles exist in MashZone NextGen? (page 42)

### 3.8.3 Which user roles exist?

The following roles exist in MashZone NextGen.

The different roles bundle various permissions.

- Admin Console User: has read-only access to the Admin Console.
- Administrator: has all available permissions in MashZone NextGen.
- Dashboard Designer: has all permissions to create and edit dashboards.
- Feed Designer: has all permissions to create and edit data feeds.

## 3.9 Install or unistall MashZone NextGen as Windows services

You can install MashZone NextGen Server as Windows services by running the **mashzonenextgen-service.bat** file.

The file is located in the <MashZone NextGen installation>\apache-tomcat\bin directory.

### Procedure

1. Run **mashzonenextgen-service.bat -install** to install MashZone NextGen Server as Windows services.
2. Run **mashzonenextgen-service.bat -remove** to uninstall MashZone NextGen Server as Windows services.

The Windows services **Software AG MashZone NextGen <version>** is installed or uninstalled.

# 4 MashZone NextGen Server Configuration

## 4.1 Memory Configuration for the MashZone NextGen Server

MashZone NextGen is initially installed with default memory settings for a small web application. Your actual memory requirements may vary significantly based on your expected load, throughput and environment.

**Note:** With release 3.7, MashZone NextGen is no longer supported on 32-bit architectures which have memory access limitations from Java.

If you are working with large datasets in MashZone NextGen Analytics or you are deploying MashZone NextGen in a staging or production environment, you may need to tune this default memory configuration. Which options you can configure depends on your usage and environment considerations:

| If | See |
|---|---|
| ▪ Your computer has less than 4G of RAM memory, or<br>▪ You have **not** installed BigMemory Max Server(s). | Configuration When MashZone NextGen Uses Only Heap Memory (page 44) |
| ▪ You have installed BigMemory Max Server(s), or<br>▪ Your computer has more than 4G of RAM memory. | Configuration When MashZone NextGen Uses Heap and Off-Heap Memory (page 45) |

### 4.1.1 Configuration When MashZone NextGen Uses Only Heap Memory

The initial default Java heap memory settings for MashZone NextGen are appropriate for small applications or development environments 2G as the maximum heap size.

If you are using MashZone NextGen with large datasets and limited memory (less than the recommended 4G minimum), configuring BigMemory Max to use off-heap memory is **not** recommended as it can adversely affect performance.

Some portion of available memory must be reserved for the operating system and any other applications on this host.

How much memory to allocate to MashZone NextGen depends on the available memory and the applications that may run on this computer.

**Procedure**

1. In a text editor of your choice, open the application server configuration file MashZoneNG-install/apache-tomcat/conf/wrapper.conf
2. Change any of these Java memory options:

   wrapper.java.initmemory, Default = **512**

   wrapper.java.maxmemory, Default = **2048**
3. Save your changes and restart the MashZone NextGen Server. See Start and Stop the MashZone NextGen Server (page 21) for instructions.

### 4.1.2 Configuration When MashZone NextGen Uses Heap and Off-Heap Memory

MashZone NextGen should be configured to use both heap and off-heap memory only when the available memory supports this adequately and you have also installed BigMemory Max Servers.

You must have installed a copy of your BigMemory Max license in MashZone NextGen to use off-heap memory. See Manage Licenses for MashZone NextGen and BigMemory Max (page 23) for instructions.

With combination heap and off-heap memory, as this figure shows, BigMemory Max uses off-heap memory for the MashZone NextGen Analytics In-Memory Stores and MashZone NextGenMashZone NextGen caches. All other MashZone NextGen processing remains in heap.

The total available off-heap memory may be limited to local off-heap memory as shown above, or it may include additional off-heap memory on external hosts if you have installed BigMemory Max Server arrays.

**Procedure**

1. In a text editor of your choice, open the application server configuration file MashZoneNG-install/apache-tomcat/conf/wrapper.conf.
2. Change or add either of these memory options used with BigMemory Max, see table below.
3. Change or set any of these Java memory options.

   wrapper.java.initmemory, Default = **512M**

   wrapper.java.maxmemory, Default = **2G**

   See the Java Tuning White Paper for more information and suggestions.

4. Save your changes and restart the MashZone NextGen Server. See Start and Stop the MashZone NextGen Server (page 21) for instructions.

| Memory options used with BigMemory Max | Description |
|---|---|
| wrapper.java.additional.<n+1><br>=-Dpresto.bm.maxOffHeap | Default = **1G**<br>Where n is the number of last additional Java parameter.<br>This is the maximum size of local off-heap memory that BigMemory Max can use for the MashZone NextGen Analytics In-Memory Stores and MashZone NextGen caches.<br>This property sets off-heap memory limits in the MashZoneNG-config/ehcache.xml configuration file. The total size of off-heap memory may include additional, external memory depending on how BigMemory Max is deployed. |
| wrapper.java.additional.<n+2><br>=-XX:MaxDirectMemorySize | Default = 1500M<br>Where n is the number of last additional Java parameter.<br>This Java memory option must be set to allow access to both off-heap and an additional allocation for Java.<br>The value of this option must always be larger than the memory allocated to off-heap. A good rule of thumb is at least 500M more. |

## 4.2 Configure feed processing time zone

MashZone NextGen feed processing represents date values in an internal format that refers to the time zone. All text to date or date to text conversions also use the configured time zone, unless an explicit time zone is specified in values or formatting functions.

The time zone is configured in the **mashzone.properties** file located in the following directory.

<MashZone NextGen installation>/apache-tomcat/webapps/mashzone/WEB-INF/

Open the **mashzone.properties** file and specify the **mashzone.feedprocessing.timezone** property.

You can only use time zones that are compatible with the Java **TimeZone.getTimeZone** method. For more details, see the corresponding Java documentation.

**Examples**

- UTC
- Europe/Berlin
- Asia/Kolkata
- GMT-5
- GMT+4:30

The default value is UTC.

This setting introduces unified time zone handling throughout feed processing. Note that this setting applies to all feed processing operators created from MashZone NextGen 10.4. Operators in feeds and dashboards that were migrated or imported from MashZone NextGen 10.3 and earlier, will run in MashZone NextGen 10.3 compatibility mode with respect to time zones. These feeds and dashboards use time zones as in MashZone NextGen 10.3 and earlier, that is, the **Runtime Info** operator creates a shifted date, and some RAQL functions use the server's local time zone.

## 4.3 Support International Character Sets and Locales

International character sets are the alphabets, characters, glyphs and other symbols used in any non-English language. Technically, this includes any non-ASCII characters. To handle these characters properly, MashZone NextGen must know the character set and how it is digitally represented - the **character encoding**.

MashZone NextGen uses the UTF-8 character encoding to handle character sets for all languages. Both UTF-8 and UTF-16 can represent any Unicode character.

Unicode defines a unique encoding for every character in world languages that are currently in active use as well as some well-known dead languages, such as ancient Greek.

Locale identifies the language used and potentially specific regional spelling or usage aspects of the language, such as differences between American English (EN_us) versus Australian English (EN_au). Locale also identifies the formats used to present dates, times and numbers for that region.

Both the character encoding and locale help to ensure that MashZone NextGen properly handles and presents data to users. The areas of configuration involved in this support include:

- MashZone NextGen Repository: the character encoding for this repository determines what character sets users can use when they create artifacts. The timezone for the MashZone NextGen Repository also affects timestamps shown in MashZone NextGen.
- **Display options**: in most cases, date, time and numeric data are shown to users based on browser settings or a default locale. Some views allow users to choose date and time formats.
- See Date, Time and Numeric Display Options (page 49) for details.
- **Logging**: you can also support international characters and different locales for the messages sent to MashZone NextGen logs. See Message Log and Default Locales (page 49) for details.

### 4.3.1 Set the Repository Character Encoding

The character encoding for the MashZone NextGen Repository is defined when you create the database that will host the repository. To support international character sets, this should be set to UTF-8, or for some databases UTF-16.

**Important:** Because of known issues, artifact names and the IDs that are generated from these names are restricted to ASCII characters. The syntax and encoding names you must use are specific to each database. For more information, see:

- Documentation for your database
- And either:

  Move MashZone NextGen repository to Microsoft SQL Server (page 25)

  Move the MashZone NextGen repository to MySQL (page 27)

  Move the MashZone NextGen repository to Oracle (page 29)

  Move the MashZone NextGen repository to PostGres (page 30)

### 4.3.2 Set the Repository Timezone or Offset

The default timezone that is used to record timestamps such as the created date and time for an artifact is the timezone for the host of the MashZone NextGen Repository. You can change the timezone used to save repository timestamps or set an offset so that repository timestamps are displayed in a different timezone:

- Force the timezone that the MashZone NextGen Repository usesto match the timezone for the MashZone NextGen Server. See Synchronize the MashZone NextGen Repository and MashZone NextGenMashZone NextGen Server Time Zones (page 146) for instructions.

- Configure the display timezone in MashZone NextGen as an offset from UTC. Note that this does not affect the actual timezonerecorded in the MashZone NextGen Repository.
- This offset is defined in the **repositoryTimezoneOffset** property in web-apps-home/mashzone/hub/config.jsfile. This property is undefined by default.
- Edit this JSON property, setting the number of minutes as a UTCoffset. For example, **300** sets this to Eastern Standard Timewhile **-180** sets this to Arabic Standard Time.
- Once the property is saved, restart the MashZone NextGen Server.

### 4.3.3 Date, Time and Numeric Display Options

In general, MashZone NextGen displays dates, times and numeric data using the formats defined by the browser's locale for the current user. If no locale information is available from the browser, MashZone NextGen use the default system locale which typically is EN_us.

Some built-in views in MashZone NextGen, allow users to choose a date and time pattern for result data such as **mm/yyyy**, for the month and year, or **EEE MMM dd, yyyy**, for the day of the week, month name, day and year. This pattern determines the widgets of dates or times that display in that view, but the language, order and delimiters used in the display are determined by the user's locale or the default locale.

### 4.3.4 Message Log and Default Locales

MashZone NextGen uses the default locale defined for the JVM for all messages that are added to MashZone NextGen logs. These defaults may also be used as the locale for artifacts if no locale is defined by users or provided by the client browser.

**To set the locale for the JVM for MashZone NextGen using Java properties**

- In a text editor of your choice, open the file **<MashZoneNG-installation>/apache-tomcat/conf/wrapper.conf**.
- Add the lines just below the line which sets the last additional Java parameter:
- wrapper.java.additional.<n+1>=-Duser.county=country-code
- wrapper.java.additional.<n+2>=-Duser.language=language-code
- Where n is the number of last additional Java parameter.

For example: **wrapper.java.additional.20=-Duser.county=CA**
**wrapper.java.additional.21=-Duser.language=fr**

- Save your changes to the file.

- Restart the MashZone NextGen Server. See Start and Stop the MashZone NextGen Server (page 21) for instructions.

## 4.4 Configure the MashZone NextGen server with custom ports

Port configuration is initially set when you install MashZone NextGen. If you change these ports or you need to host multipleMashZone NextGen Servers on one host, you must update configuration in theMashZone NextGen Server. You may also need to change ports for the MashZone NextGen Repository and for Tomcat.

### 4.4.1 Change MashZone NextGen Server Ports

The host name and port for theMashZone NextGen Server defaults to localhost and 8080 respectively. The port is typically defined in configuration for Tomcat, the application server that hosts MashZone NextGen.

**Procedure**

1. Update port configuration for the application server that hostsMashZone NextGen in MashZoneNG-install/apache-tomcat/conf/server.xml in the <Connector> elements for HTTP and/or HTTPS.
2. Restart the MashZone NextGen Server. See Start and Stop the MashZone NextGen Server (page 21) for instructions.

### 4.4.2 Change MashZone NextGen Repository Ports

If you are running multiple instances of the MashZone NextGen Server in one host with separate instances of the MashZone NextGen Repository, you may need to use different ports for each database instance.

- The default MashZone NextGen Repository. No updates to ports are needed as the Derby database is embedded.
- The MashZone NextGen Repository is hosted in a robust database. You must use different ports for each MashZone NextGen Repository instance.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **MashZone NextGen Repositories** section.

3. Select **Metadata Repository** and:
   a. Change the JDBC URL to the correct host and port number.
   b. Click **Save**.
4. Restart the MashZone NextGen Server. See Start and Stop the MashZone NextGen Server (page 21) for instructions.

### 4.4.3 Tomcat Application Server Port

If you are running multiple instances of the MashZone NextGen Server in one host, you must have separate application server instances for each. You must update the following ports:

| Configuration File | Element |
|---|---|
| MashZoneNG-install/apache-tomcat/conf/server.xml | **<Server>** to set the command port<br>**<Connector> for the HTTP port**<br>**<Connector> for the HTTPS port** |

## 4.5 Configure the MashZone NextGen server to work with a proxy server

MashZone NextGen is **only** compatible with HTTP proxy servers.

If you have a proxy server in your environment that MashZone NextGen should use, you **must** add configuration information to the MashZone NextGen Server for the proxy server. You can also define a **white list** of addresses that do not require proxy server access. See Define a Proxy Server White list for MashZone NextGen (page 54) for more information.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. In the Server section, click **Proxy Settings**.
3. Set the **Enable Proxy** option.
4. Set the following connection properties for your proxy server:

   **Host** = the host name or IP address for the proxy server. This is required.

   **Port** = the port number for the proxy server. This is required.

   **Username** = the user name that the MashZone NextGen Server should use to connect to the proxy server. This is only required if your proxy server requires credentials.

**Password** = the password that the MashZone NextGen Server should use to connect to the proxy server. This is only required if your proxy server requires credentials.

5. If needed, define a white list of addresses that should not use the proxy server. See Define a Proxy Server White list for MashZone NextGen (page 54) for instructions.
6. Click Save proxy settings.

## 4.6 Embedding MashZone NextGen in external system environments

You can use MashZone NextGen as a widget in external products, for example, webMethods Business Console. As embedded widget MashZone NextGen is enabled to send data via outbound API (Post data) to the embedding system and receive data via inbound API (URL selection) from the embedding system.

### 4.6.1 Configure MashZone NextGen server to work with iFrame

By default, MashZone NextGen can be embedded using HTML inline frames (iFrame) if the MashZone NextGen server and the server of the embedding system use the same protocol, same host and same port.

To embed MashZone NextGen within another HTML document the iFrame source points to the MashZone NextGen dashboard as shown in the following example. **<iframe id="embedded-mzng-dashboard" width="600px" height="600px" src="http://mzngServerHost:mzngServerPort/mashzone/hub/dashboard/dashboard.jsp?mzngDashboardGUID"> Your browser does not support iframes. </iframe>**

If the embedding system is running on a different host or uses a different protocol or port, the MashZone NextGen server must be configured as follows. The MashZone NextGen server configuration file **applicationContext-security-filters.xml** needs to be configured by adding filters for **X-Frame-Options** and content security policies.

The **applicationContext-security-filters.xml** server configuration file is located in following directory. <MashZone NextGen-installation>/apache-tomcat/webapps/mashzone/WEB-INF/classes.

**Procedure**

1. Open the applicationContext-security-filters.xml configuration file in a text editor of your choice.

2. Adapt the security settings as follows and exchange the string "http://otherServerHost:otherServerPort" with the system origin MashZone NextGen is to be embedded in.

```
<beans:beans
xmlns="http://www.springframework.org/schema/security"...> ... <http
pattern="/hub/(login|reset_password)\.html.*" security="none"
request-matcher="regex"/> <http pattern="/help/.*" security="none"
request-matcher="regex"/> <http pattern="/**/*.jsp"
use-expressions="false"
authentication-manager-ref="authenticationManager"
entry-point-ref="mzngAuthenticationEntryPoint"> <anonymous
enabled="false"/> <headers> <!--frame-options policy="SAMEORIGIN"/-->
<frame-options policy="ALLOW-FROM" strategy="static"
value="http://otherServerHost:otherServerPort" />
<!--content-security-policy policy-directives="frame-ancestors
'self'"/--> <content-security-policy policy-directives="frame-ancestors
'self' http://otherServerHost:otherServerPort"/> </headers> <csrf
token-repository-ref="csrfTokenRepository"
request-matcher-ref="skipHttpAuthCsrfMatcher"/> <custom-filter
ref="samlTokenProcessingFilter" after="PRE_AUTH_FILTER"/>
<custom-filter ref="jwTokenProcessingFilter" before="CAS_FILTER"/>
<custom-filter ref="credentialContainerFilter"
before="EXCEPTION_TRANSLATION_FILTER"/> </http> <http
pattern="/**/*.html" use-expressions="false"
authentication-manager-ref="authenticationManager"
entry-point-ref="mzngAuthenticationEntryPoint"> <intercept-url
pattern="/**/*.html" access="IS_AUTHENTICATED_ANONYMOUSLY"/> <anonymous
enabled="false"/> <headers> <!--frame-options policy="SAMEORIGIN"/-->
<frame-options policy="ALLOW-FROM" strategy="static"
value="http://otherServerHost:otherServerPort" />
<!--content-security-policy policy-directives="frame-ancestors
'self'"/--> <content-security-policy policy-directives="frame-ancestors
'self' http://otherServerHost:otherServerPort"/> </headers> </http> ...
</beans:beans>
```

3. Save changes.

Your changes will be applied with the next MashZone NextGen server start.

Further details on the topic **Using iFrame** can be found in the spring security documentation: https://docs.spring.io/spring-security/site/docs/current/reference/html/headers.html#headers-frame-options.

## 4.6.2 Post data

The **Post data** action creates an outbound API to pass data from MashZone NextGen dashboards to an embedding system, for example, an external web application. The action is available for most widgets.

### 4.6.3 URL selection

With the **URL selection** MashZone NextGen provides an inbound API to receive data from an embedding system, for example, an external web application. The action is available for most widgets.

## 4.7 Define a Proxy Server White list for MashZone NextGen

If you have configured a proxy server for the MashZone NextGen Server, you can define a white list of domains, hosts or IP addresses that do **not** require access through the proxy server.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. In the **Server** section, click **Proxy Settings**.
3. Set the following properties:

   **Bypass IP List** = enter one or more IP address, separated by commas, that the MashZone NextGen Server should access without the proxy server. To use wildcards in IP addresses, see Using Regular Expressions in a White list (page 54).

   **Bypass Host List** = enter one or more fully-qualified host names, separated by commas, that the MashZone NextGen Server should access without the proxy server.To use wildcards in IP addresses, see Using Regular Expressions in a White list (page 54).

   **Bypass Domain List** = enter one or more domain names, separated by commas, that the MashZone NextGen Server should access without the proxy server. To use wildcards in IP addresses, see Using Regular Expressions in a White list (page 54).
4. Click **Save proxy settings**.
5. Log out of MashZone NextGen.
6. Stop and restart the MashZone NextGen Server to apply these changes.

### 4.7.1 Using Regular Expressions in a White list

All of the white list properties accept **regular expressions** to define sets of IP, host or domain name addresses using wildcards. You can use any valid regular expression character, however, the most common wildcard characters that you may use are:

| | |
|---|---|
| Replace Any Single Character | . |

| | |
|---|---|
| Replace Any Number of Characters | **.*** |

Examples and solutions for the most common patterns include:

- Specifying Literal Dot Separators (page 55)
- Specifying Domains (page 55)
- Specifying Host Names (page 56)

## 4.7.2 Specifying Literal Dot Separators

Because the dot character is used as a wildcard in regular expressions and is also the standard separator for groups in IP addresses, domain names and host names, you can get unintended results when using wildcards. For example, this is a valid regular expression for IP addresses:

**139.16.1.***

On Windows systems, many administrators would expect this to expression to "match any IP address with first-through-third groups of 139, 16 and 1 respectively" such as **139.16.1.10** and **139.16.1.35**.

This would actually match either of these IP addresses:

**139.16.1.10**

**139.16.11.120**

In most cases, the difference between a literal dot and the dot as a wildcard character doesn't make a difference. If you need to clarify a white list entry to match a literal dot, use **\.** instead.

The expression **139\.16\.1\..*** would correctly match **139.16.1.10** and **139.16.1.35** but would not match **139.16.11.120**. In many cases, you could also simplify this to **139.16.1\..*** to get the correct behavior.

## 4.7.3 Specifying Domains

With domains, you must specify a wildcard at the beginning of the domain name. This example is not a valid domain name expression:

**mydomain.com**

This entry would **not** match a host name of **east.mydomain.com** or **east.customers.mydomain.com**. To specify the domain correctly, enter:

**.*mydomain.com**

### 4.7.4 Specifying Host Names

In white list properties, host names are fully-qualified. Thus **stives** is not a valid host name while **stives.customers.mydomain.com** is valid. A host name expression of **.*customers.mydomain.com** would match all of these hosts:

**stives.customers.mydomain.com**

**cour.customers.mydomain.com**

**tempcustomers.mydomain.com**

Note that an expression of **.*.customers.mydomain.com** would also match these same three hosts.

You may need to specify literal dot separators in host names also to properly clarify the expressions. If in this example you did **not** want **tempcustomers.mydomain.com** to be matched, you would need an expression such as **.*\.customers.mydomain.com**. See Specifying Literal Dot Separators (page 55) for more information.

## 4.8 Configure MashZone NextGen for TLS and Digital Certificates

MashZone NextGen expects HTTP as the default transport protocol from clients to the MashZone NextGen Server.

MashZone NextGen supports HTTPS and TLS for connections from clients or connections to many types of information sources.

The certificate store, certificates and configuration needed to support TLS in MashZone NextGen depends on the connection requirements, as shown below:

| | Certificate Store and Certificates | | Store Configuration | | | MashZone NextGenMashZone NextGen Configuration | |
|---|---|---|---|---|---|---|---|
| | Key | Trust | Java | App Server | MashZone NextGen MashZone NextGen | Authentication | Security Profiles |
| One-Way TLS to MashZone NextGen | ✓ | | | ✓ | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| (page 58). | | | | | | | |
| Mutual to MashZone NextGen<br>See Configure Mutual TLS Between Users and MashZone NextGen (page 58). | ✓ | ✓ | | ✓ | | ✓ | |
| One-Way TLS to Mashable Information Sources (page 59). | | ✓ | can be in either | | | | |

See the chapter The Certificate Store and Certificates (page 57) for more information.

### 4.8.1 The Certificate Store and Certificates

Both key stores and trust stores are **certificate stores** to store and manage the **key certificate pairs** or **public certificates** used in secure connections with the TLS protocol. Key stores manage key certificate pairs and trust stores manage the public certificates of trusted peers.

### 4.8.2 Key Certificate Pairs

For MashZone NextGen, the key certificate pair stored in the key store identifies the MashZone NextGen Server to users, for both one-way and mutual TLS. The key certificate pair identifies the MashZone NextGen Server to information sources for mutual TLS.

You must generate a key certificate pair for MashZone NextGen. Typically you also have the key certificate pair signed by a Certificate Authority and import this into the certificate store using the Java **keytool** utility or other certificate management tools.

### 4.8.3 Trusted Peer Certificates

The public certificates from peers are stored in the trust store and identify users, for mutual TLS, or identify information sources, for one-way or mutual TLS.

When public certificates for peers are signed by well known Certificate Authorities, they are automatically verified and imported into the trust store. If public certificates are self-signed or signed by an unknown Certificate Authority (the CA root certificate is not found in the trust store), you must obtain and import the public certificates to the trust store before the first connection occurs during user login.

### 4.8.4 The Certificate Store

You can use a single certificate store as both the key store and trust store for MashZone NextGen or you can use separate certificate stores. You can use an existing certificate store for MashZone NextGen, such as the default certificate store shipped with some application servers. Or you can create a new certificate store using the Java **keytool** utility.

See Java keytool documentation (http://docs.oracle.com/javase/6/docs/technotes/tools/solaris/keytool.html) for more information, commands and instructions on managing key certificate pairs, trusted certificates and certificate stores.

### 4.8.5 Configure Mutual TLS Between Users and MashZone NextGen

The MashZone NextGen Server and users both exchange certificates. MashZone NextGen can also be configured to use user digital certificates for authentication. The connection requires:

- Store and Certificates:

  A certificate store as key store and trust store for the MashZone NextGen Server.

  A key certificate pair for the MashZone NextGen Server.

  Public certificates in the trust store for any user public certificates that are self-signed.

  You must add self-signed certificates to the trust store before these users login. See Trusted Peer Cerficates (page 57) for more information.

  See The Certificate Store and Certificates (page 57) for more information.
- Configuration in the application server hosting MashZone NextGen to use the HTTPS port. This also includes configuration identifying the key store and trust store for the MashZone NextGen Server. See Configure HTTPS and Certficate Stores in the Application Server (page 59) for instructions.

### 4.8.6 One-Way TLS to MashZone NextGen

This requires:

1. A key store and a key certificate pair for MashZone NextGen. See The Certificate Store and Certificates (page 57) for more information.
2. Configuration in your application server for the HTTPS port to MashZone NextGen and the key store. See Configure HTTPS and Certficate Stores in the Application Server (page 59) for instructions.

### 4.8.7 One-Way TLS to Information Sources

This requires:

- A trust store for MashZone NextGen. See The Certificate Store and Certificates (page 57) for more information.

  Configuration for the trust store in either:

  The application server hosting the MashZone NextGen Server. See Configure HTTPS and Certficate Stores in the Application Server (page 59) for instructions.
- Java. See Update TLS Configuration for Java (page 60) for instructions.
- Self-signed certificates, if any, for data source using one-way TLS. You must add these certificates to the trust store before the data source can be registered. See Trusted Peer Cerficates (page 57) for more information.

### 4.8.8 Configure HTTPS and Certificate Stores in the Application Server

Configuration for TLS for MashZone NextGen can be defined in the application server that hosts the MashZone NextGen Server. These instructions discuss the basic steps for configuring TLS in Tomcat. See the official Tomcat documentation or the documentation for your application server for detailed information.

**Procedure**

1. If you do not yet have a key store, trust store and certificate for the MashZone NextGen Server, find or create these stores and certificate. See the Certificate Store and Certificates (page 57) for instructions.
2. Configure Tomcat for secure connections from clients to the MashZone NextGen Server.

   We recommend using TLSv1.2 for network communication.

   **Warning**

   We recommend not using TLSv1.0 and TLSv1.1 in your production environments, as TLSv1.0 and TLSv1.1 are considered unsafe for network communication.

   You can add the **sslEnabledProtocols** and **sslProtocols** parameters to the **server.xml** file to disable TLSv1.0 and TLSv1.1. The parameters allow limiting the supported protocols. For details, see the official Tomcat documentation.

   a. Edit the server.xml file for Tomcat to uncomment and configure the <Connector> element for TLS/HTTPS 1.1. For example:

   ```
   <Connector port="8443"
   protocol="org.apache.coyote.http11.Http11Nio2Protocol"
   ```

```
SSLEnabled="true" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" disableUploadTimeout="true"
acceptCount="100" scheme="https" secure="true"
clientAuth="true" sslProtocol="TLS" keystoreFile="conf/tomcat.jks"
keystorePass="keystrpwd" truststoreFile="conf/tomcat.jks"
truststorePass="truststrpwd" />
```

This example uses the default Tomcat port, 8443, and mutual TLS, based on the **clientAuth** value. If this was a one-way connection, you would set **clientAuth** to false. This example also uses the default Tomcat certificate store, **conf/tomcat.jks**, as both the key store and the trust store. See Tomcat documentation for information on other properties.

b. Once you have configured an HTTPS port in your application server, update port configuration for the MashZone NextGen Server to listen to that port. See Configure the MashZone NextGen server with custom ports (page 50) for more information on this step.

c. Enable MashZone NextGen to use secure session cookies:

   Open the **web.xml** file located in **<MashZone NextGen installation>/apache-tomcat/webapps/mashzone/WEB-INF/** in a text editor.

   Find the **session-config/cookie-config/secure** element and change the value to **true**.

   Example

```
<session-config> <session-timeout>30</session-timeout> <!-- Set the
"secure" flag to true when using HTTPS for enhanced security -->
<cookie-config> <secure>false</secure> </cookie-config>
</session-config>
```

   Once this is set to true, only HTTPS access will be allowed.

## 4.8.9 Update TLS Configuration for Java

For the data acquisition using HTTPS, certificates for secure endpoints are validated against the default trust store for Java (JRE).

For more information on the default JRE trust store, see http://download.oracle.com/javase/6/docs/technotes/guides/security/jsse/JSSERefGuide.html#CustomizingStores (http://download.oracle.com/javase/6/docs/technotes/guides/security/jsse/JSSERefGuide.html#CustomizingStores).

Initially, this may not be the trust store you have configured for the MashZone NextGen Server in the application server. This can cause security errors for information sources.

To avoid these errors, you can configure the JRE to use the trust store for the MashZone NextGen Server.

**Procedure**

1. Open the application server configuration file MashZoneNG-install/apache-tomcat/conf/wrapper.conf in a text editor of your choice.
2. Add the following Java system properties:

   wrapper.java.additional.<n+1>=-Djavax.net.ssl.trustStore=/path/to/mashup-server/truststore

   This is the absolute path to the trust store for the MashZone NextGen Server.

   wrapper.java.additional.<n+2>=-Djavax.net.ssl.trustStorePassword=truststore-password

   This is only required if the MashZone NextGen Server's trust store uses a password.

   Where n is the number of last additional Java parameter.
3. Save your changes to the script and restart the MashZone NextGen Server. See Start and Stop the MashZone NextGen Server (page 21) for instructions.

## 4.9 MashZone NextGen Logging

In addition to logging from your application server, MashZone NextGen provides the following types of logging:

- Basic logging for MashZone NextGen Server startup, shutdown and exceptions based on a configured logging level. See Configure Logging for the MashZone NextGen Server (page 61) for more information.
- Audit logging for dashboards, data feeds, aliases, and permissions. See Audit logging for dashboards, data feeds, aliases, and permissions (page 63) for details.

### 4.9.1 Configure Logging for the MashZone NextGen Server

The MashZone NextGen Server log, **wrapper.log**, logs all exceptions from startup through shutdown. See MashZone NextGen Logging (page 61) for links to additional types of logging you can use with MashZone NextGen.

For clustered environments, updating logging configuration affects logging only for the MashZone NextGen Server where you are currently logged in. Generally, this is the behavior you want.

To change logging for all MashZone NextGen Servers in the cluster, update logging configuration for one server and copy the updated

MashZoneNG-install/apache-tomcat/conf/log4j.properties file to each of the other MashZone NextGen Servers in the cluster. You do not need to restart MashZone NextGen Servers.

### Procedure

1. Open the **log4j2-mashzone.xml** configuration file in <MashZone NextGen installation>/apache-tomcat/conf/ with an text editor of your choice.
2. Edit any of the following properties:

   **Root Log Warning** = the general logging level to use, such as **ERROR**. All exceptions for that level and above will be logged, so this contributes directly to how quickly logs may grow. **DEBUG** is the most verbose logging level.

   **Log file path** = both the folder where the log files for the MashZone NextGen Server should be saved and the name to use for log files. This defaults to tomcat-install/logs/prestoserver.log.

   You can use an absolute path or a relative path. Relative paths are relative to the web-apps-home folder.

   **Maximum log file size** = maximum size for a log file. Once a file has reached this size is it saved as a numbered backup, such as **MashZone.log.1** and a new log file is started.

   **Data nucleus logging level** = This property should only be changed when requested by MashZone NextGen technical support to help debug specific issues. It is the logging level to use in the data mapping layer for MashZone NextGen.

   **HTTP client logging level** = This property should only be changed when requested by MashZone NextGen technical support to help debug specific issues. It is the logging level to use for requests/responses between the MashZone NextGen Server and information sources.

   **NET SF logging level** = This property should only be changed when requested by MashZone NextGen technical support to help debug specific issues. It is the logging level to use for JSON serialization and deserialization.

   **ACEGI security logging level** = This property should only be changed when requested by MashZone NextGen technical support to help debug specific issues. It is the logging level to use with the MashZone NextGen security layer.

   **Apache logging level** = This property should only be changed when requested by MashZone NextGen technical support to help debug specific issues. It is the logging level to use with many of the third party libraries used in MashZone NextGen.

**Spring framework logging level** = This property should only be changed when requested by MashZone NextGen technical support to help debug specific issues. It is the logging level to use for server initialization, shutdown and the request/response pipeline for the MashZone NextGen Server.

3. If desired, click **Advanced Options** to set any of these properties:

   **Log class for normal logging** = the java class that handles appending log entries to the log file. This defaults to **org.apache.log4j.RollingFileAppender**.

   **Layout class for normal logging** = the Java class that handles the layout pattern for entries to log files. This defaults to **org.apache.log4j.PatternLayout**.

   **Layout pattern for normal logging** = the expression defining the pattern for entries to the log file. This defaults to **%d %p [%c** - %m%n

   **Maximum normal log file backups** = how many log backups are kept. This defaults to seven.

   The advanced properties are defined by Log4J, the underlying logging framework for MashZone NextGen. For more information, see http://logging.apache.org/log4j/1.2.

This change becomes effective automatically within 60 seconds.

## 4.9.2 Audit logging for dashboards, data feeds, aliases, and permissions

The audit logging tracks various events concerning dashboards, data feeds, aliases, and permissions. Logins, logouts and failed logins are also tracked. This logging is enabled by default.

The following events are logged.

- create, edit, and delete of dashboards
- create, edit, and delete of data feeds
- create, edit, and delete of aliases
- edit permissions
- logins, logouts and failed logins

The tracked events are logged in the **MashZone-AuditLog.log** file. The file is located in the following directory.

<MashZone NextGen installation>/logs

The audit logging can be switched on and off in the **log4j2.xml** file. You can edit the XML file with any text or XML editor. If you want to disable the audit logging, set the value of the **level** parameter to **OFF**. The default value is **info**.

The file is located in the following directory.

<MashZone NextGen installation>/conf

## 4.10 BigMemory Max for Caching, Connections and In-Memory Stores

By default MashZone NextGen uses local memory for caching. This uses BigMemory Max as a local client that is installed with MashZone NextGen and requires only your MashZone NextGen license. In specific cases, you must also install BigMemory Max Servers on one or more additional hosts and configure MashZone NextGen and the Integrated MashZone Server to work with them. MashZone NextGen requires BigMemory Max Servers with a BigMemory Max license to support:

- Significant, extensible amounts of memory, most commonly for very large datasets used with MashZone NextGen.

  BigMemory Max Servers can be deployed in clusters, also known as Terracotta Server Arrays, that can easily be extended for scalable memory requirements.
- Distributed caching when MashZone NextGen is deployed in clusters.

  With clusters, some of the internal MashZone NextGen caches must be distributed and managed by BigMemory Max Servers.
- Access to off-heap memory

  BigMemory Max Servers also can manage memory outside of heap both for better scalability and performance improvements.
- Access to In-Memory Stores created and populated dynamically by external systems.

  BigMemory Max manages the In-Memory Stores created dynamically by other systems and makes connection information available to MashZone NextGen through the Terracotta Management Console (TMC) to allow MashZone NextGen to work with this data. Apama, for example, dynamically creates distributed stores for the Apama MemoryStore which MashZone NextGen can connect to and query.

  MashZone feeds that use BigMemory Max connections as a data source.

See Configure BigMemory Max Servers for MashZone NextGen Caching and In-Memory Stores (page 66) for instructions on configuring MashZone NextGen to work with BigMemory Max Servers.

You also need to provide configuration and connection information for In-Memory Stores that are created by other systems. For more information on these tasks, see:

- Declare BigMemory Max Stores for MashZone NextGen (page 68)
- Manage Dynamic BigMemory Max Stores for MashZone (page 73).

See Caching for the MashZone NextGen Server (page 65) for an overview of MashZone NextGen caching. For caching configuration, see Distributed Caching for MashZone NextGen Clusters (page 66).

## 4.10.1 Caching for the MashZone NextGen Server

The MashZone NextGen Server caches artifact metadata, for internal purposes. By default, caches are stored in local memory. See Artifact Caching (page 65) and Response Caching (page 65) for details.

All MashZone NextGen caches can be distributed when MashZone NextGen is deployed in clusters. See Distributed Caching for MashZone NextGen Clusters (page 66) for an overview of distributed caching for MashZone NextGen.

### 4.10.1.1 Artifact Caching

Artifact caching caches metadata for feeds and other internal operational data when they are run in MashZone NextGen.

Because updates to artifacts are typically infrequent, this cache is long-lived. It is not persisted to disk. Cache entries are flushed only when an artifact is updated or when the server hosting the cache is restarted. No additional configuration is required to enable local artifact caching for MashZone NextGen.

### 4.10.1.2 Response Caching

Response caching caches the responses from dashboards and data feeds when they are run. This is a short lived cache that caches response data based on the unique signature of each request to dashboards and data feeds.

Configuration for response caching gives you fine grained control for which dashboards and data feeds use response caching and the expiration periods for their cache entries.

### 4.10.1.3 Distributed Caching for MashZone NextGen Clusters

When MashZone NextGen is deployed in clusters, artifact caching must be distributed to maintain cache integrity. Response caching, however, can be left in local memory for each MashZone NextGen Server or it can be distributed.

In many environments, local caching provides both good performance and acceptable cache integrity for response caching. Local caching is "eventually consistent", but can result in visible differences as cached responses are not guaranteed to be identical for different cluster members. For environments that cannot tolerate any loss of cache integrity, distributed response caching is recommended.

Distributed caching is only available if you purchase and deploy BigMemory Max Servers.

You use BigMemory Max Servers to handle distributed caching for MashZone NextGen.

With BigMemory Max Servers, data for most MashZone NextGen caches can use the total off-heap memory configured for the cluster plus any heap and off-heap memory configured for the MashZone NextGen local host.

The BigMemory Max Servers manage consistency and memory across the cluster. They also support failover, with mirror servers, for high availability and many other advanced capabilities that may be useful for enterprise production systems.

To configure distributed caching, see Configure BigMemory Max Servers for MashZone NextGen Caching (page 66) for set up instructions.

### 4.10.1.4 Configure BigMemory Max Servers for MashZone NextGen Caching

You can configure MashZone NextGen to work with one or an array of BigMemory Max Servers to provide additional memory, provide reliability and support specific other features. See BigMemory Max for Caching, Connections and In-Memory Stores (page 64) for more information on features that require BigMemory Max servers.

**Procedure**

1. Copy your license for BigMemory Max to MashZone NextGen and update MashZone NextGen startup scripts:
   a. Copy the license file terracotta.key to the MashZoneNG-install/apache-tomcat/conf folder.

      If MashZone NextGen is deployed in a cluster, you must copy this file to every cluster member.

b. Add the following Java system property to the MashZone NextGen server configuration file <MashZone NextGen installation>/apache-tomcat/conf/wrapper.conf:

**wrapper.java.additional.<n+1>=Dcom.tc.productkey.path=MashZoneNG-install/apache-tomcat/conf/terracotta.key**

Where n is the number of last additional Java parameter.

If MashZone NextGen is deployed in a cluster, you must update the server configuration files for every cluster member.

2. Edit the **ehcache.xml** file for MashZone NextGen in a text editor of your choice.
3. Find the line in ehcache.xml with <terracottaConfig> that is commented out like this:

   **<!-- <terracottaConfig url="localhost:9510" /> -->**

   Remove the comment markers and change the **url** attribute to the host (or IP address) and port for the BigMemory Max server(s) you installed. For example:

   **<terracottaConfig url="tcHost1:9510" />**

   There are several ways to identify one or more BigMemory Max servers for MashZone NextGen. See BigMemory Max documentation (http://terracotta.org/documentation/) for more information.
4. Find the line in ehcache.xml with <terracotta> that is commented out and uncomment this line for each of the following named <cache> elements:

   **SEARCH_RESULTS_CACHE** = one part of the MashZone NextGen Artifact cache.

   **SERVICES_BY_ID_CACHE** = = one part of the MashZone NextGen Artifact cache.

   **SERVICE_RESPONSE_CACHE** = the MashZone NextGen Response cache for dashboards and data feeds. This is optional. Update this cache **only** if you want it to be distributed.

   This **<terracotta>** element allows the In-Memory Store and MashZone NextGen caches to use heap and off-heap memory in both the local host and BigMemory Max hosts. This combined memory is managed by BigMemory Max.

   For more information on the **<terracotta>** element, see **Distributed Configuration** topics in BigMemory Max documentation (http://terracotta.org/documentation/).
5. Save these changes to ehcache.xml.

   For clusters where this configuration file is not stored in a shared external folder, copy this file to the same location for each MashZone NextGen cluster member.
6. Start BigMemory Max Server(s).
7. If needed, adjust memory configuration for the local MashZone NextGen host. See Memory Configuration for the MashZone NextGen Server (page 44) for instructions.

8. Restart MashZone NextGen. See Start and Stop the MashZone NextGen Server (page 21) for instructions.

### 4.10.2 Working with BigMemory Max Stores for RAQL

Declaring In-Memory Stores for MashZone NextGen allows MashZone NextGen to connect to stores in BigMemory Max that may have been created and populated by other systems. These In-Memory Stores allow you to store large datasets for quick access and to retrieve them for analysis using RAQL.

Each In-Memory Store holds one dataset with any number of rows, also known as entries, within the memory constraints defined for the store. In previous releases you could store many different datasets in one store with a dataset being one entry in the store.

To allow RAQL to work with external data, external memory stores must be configured with:

- The name of the cache manager that manages memory for the in-memory store.
  Note: BigMemory Max does not require cache manager names, but they are a best practice for caches used as RAQL In-Memory Stores. Cache manager names prevent potential name collisions for stores.
- Search attributes for the dataset that identify the columns in the dataset and the datatype of the data in each column.

There are two types of In-Memory Stores that differ in the way they are configured in MashZone NextGen: declared and dynamic In-Memory Stores.

#### 4.10.2.1 Declared In-Memory Stores

Declared stores are defined in BigMemory Max configuration files (ehcache.xml files) that MashZone NextGen administrators add to the MashZone NextGen Server to allow you to work with these stores.

Declared In-Memory Stores can contain data loaded by external systems. With declared in-memory stores, the producer system that stores data in the store is the system that creates the store. MashZone NextGen is always the consumer system that retrieves data from the store for analysis using RAQL.

Declaring stores before use allows you to:

- Customize store properties for each store, providing better memory management and data retention.
- Define search attributes for the columns in the dataset in each store.

- Column search attributes are required to allow RAQL to work with datasets stored by external systems.
- Defining search information also allows RAQL to delegate filtering and sorting to the In-Memory Store which provides better performance.

Configuration for declared stores specifies the cache manager, search attributes and connection information for the store. For an example, see Declare a New In-Memory Store (page 69).

#### 4.10.2.1.1 Declare a new In-Memory Store

Define configuration for one or more In-Memory Stores in a cache configuration file for BigMemory Max (an ehcache.xml file).

It is a best practice to change the default file name **ehcache.xml** for this configuration file to something more meaningful, such as **myCRM-cache.xml**. This makes it easier to identify when multiple configuration files are uploaded to MashZone NextGen.

**Procedure**

1. Add a name attribute to the **<ehcache>** element and assign a unique name.

   This is the cache manager name which must be unique for this MashZone NextGen Server. It should consist solely of letters, numbers, underscores(_) or dashes (-).

2. Add a **<cache>** element for each store you need to declare. The following example shows some common properties. See BigMemory Max documentation (http://terracotta.org/documentation/) for more information.

   You can find this example configuration file, **sample-cache.xml**, for declared stores in the <MashZone NextGen installation> /prestocli/raql-samples folder.

   <ehcache name="sample-cache" > <diskStore path="java.io.tmpdir"/> ... <cache name="StocksDeclCache" maxBytesLocalHeap="50M" memoryStoreEvictionPolicy="LRU" timeToIdleSeconds="0" timeToLiveSeconds="0"> </cache> ... </ehcache>

   If this In-Memory Store will be populated by external systems with datasets that are Java objects, add <searchable> to the <cache> element and define a <searchAttribute> with the name, datatype and extractor class for each property in these Java objects that will contain data.

   For the **class** attribute, use the **net.sf.ehcache.search.AttributeExtractor** interface provided in the BigMemory Max Search API or an implementation class of **AttributeExtractor**. See BigMemory Max documentation (http://terracotta.org/documentation/) for details.

MashZone NextGen is only able to access searchable attributes of datasets stored by external systems. For Apama used as external system, search attributes are no more required.

Since version 9.9 MashZone NextGen supports the native Apama RowValue format. MashZone NextGen can consume RowValues stored by Apama and convert them into the RAQL record format. In case of caches written by Apama searchable attributes are no more needed for accessing the data at all but they are still required for processing filters, aggregations and sorting directly in BigMemory Max.

3. Save this file.
4. Copy the JAR file containing the classes used as search attributes to extract data from the dataset in this cache to MashZoneNG-config/lib.

   See documentation for the external system that created this dynamic store to determine what JAR files are needed. For Apama, see documentation on the MemoryStore.

   The default location for this folder in MashZone NextGen is <MashZone NextGen> installation/apache-tomcat/mashzone/WEB-INF/lib. If MashZone NextGen is deployed in a cluster, however, this location may be a separate external folder. For more information, see Setting Up an External MashZone NextGen Configuration Folder (page 111).
5. Restart the MashZone NextGen Server. For instructions, see Start and Stop the MashZone NextGen Server (page 21).
6. Open the MashZone NextGen Admin Console.
7. Click **Server** to expand this section of the **Administration** menu.
8. Click **BigMemory Max**.
9. Open the **BigMemory Max Cache** tap.

   The **BigMemory Max Cache** tab lists any In-Memory Store configuration files that have already been upload.
10. Click **Register a new EhCache Configuration File**.
11. Enter the name assigned to <ehcache> in this configuration file (in step 1) as the BigMemory Max Data Source Name. This name is used as a prefix for all stores defined in this configuration file to uniquely identify each store.

    Data source names **must** be unique for this MashZone NextGen Server. They should contain only letters, numbers, underscores (_) or dashes (-).

    If any of the declared In-Memory Stores for this connection have data populated by external systems, the data source name **must also** match the name assigned to the **<ehcache>** element in the configuration file.
12. Click **Browse** to find and select the Cache Configuration File ehcache.xml you created in step 1.

13. Click **Add this file**.

The file is uploaded to the MashZone NextGen Repository in the standard path bigmemory/caches/file-name and shown in the list by data source name. The URL shown is the relative path in MashZone NextGen to this resource.

Administrators can also manage resources files in the Admin Console. See Manage Files for MashZone NextGen Features or Artifacts (page 86) for more information.

#### 4.10.2.1.2 Modify a Declared In-Memory Store

1. Update the configuration file for a declared In-Memory Store as needed.

For example, you may need to add configuration to allow an In-Memory Store to use memory in external BigMemory Max hosts when you add servers or deploy MashZone NextGen in staging or production environments.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Server** to expand this section of the Administration menu.
3. Click **BigMemory Max**.
4. Open the **BigMemory Max Cache** tap.
5. The **BigMemory Max Cache** tab lists any In-Memory Store configuration files that have already been upload.
6. Select the existing BigMemory Max data source for this store and click **Delete**.
7. Add this data source with the updated configuration file. See Declare a New In-Memory Store (page 69) for instructions.

#### 4.10.2.1.3 View Details for Declared In-Memory Stores

You can view configuration information for declared In-Memory Stores.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Server** to expand this section of the Administration menu.
3. Click **BigMemory Max**.
4. Open the **BigMemory Max Cache** tap.

   This lists any configuration files for declared In-Memory Store that have already been upload.

5. Click the title for a configuration file to see detailed information for the In-Memory Stores declared in that file.
6. To see the configuration file contents, click the URL for that file.

#### 4.10.2.1.4 Define permissions for declared In-Memory Strores

You can define permissions for declared In-Memory Stores.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Server** to expand this section of the **Administration** menu.
3. Click **BigMemory Max**.
4. Open the **BigMemory Max Cache** tap.

   The tab lists all available configuration files for declared In-Memory Store that have been upload.
5. Click the  **Edit permission** icon to configure the permission of an existing EhCache configuration file.
6. Enter a term in the search field and click **Search**. Clicking on **Search** without any input values fetches all users and groups.
7. Drag a user or user group from the **Search results** field and drop it into the **Principals with permissions** field.

   By default, the owner of the EhCache configuration file is already present in the **Principals with permissions** list. This owner is non editable and cannot be removed from the list.
8. Enable or disable the **View** or **Usage** privileges of a user or user group.
9. Click **OK**.

Your settings are applied.

### 4.10.3 Dynamic In-Memory Stores

Dynamic stores are created programmatically on the fly when:

- An external system creates a store dynamically in BigMemory Max. These stores are also known as dynamic external stores because the external system is also the system the stores data in the store.
  Note: Access to dynamic external stores requires BigMemory Max be installed as a server or server array. MashZone NextGen must also have access to the BigMemory Max license.
- To connect to dynamic external stores, a MashZone NextGen administrator must define connection configuration. See Add an External Dynamic In-Memory Store Connection (page 453) for more information. The Terracotta Management Console (TMC) must also be running to successfully connect to a dynamic external in-memory store.
- RAQL uses the connection configuration to retrieve configuration for the store from the Terracotta Management Console (TMC) that manages the BigMemory Max host for this store. This includes search attribute information that is required to allow RAQL to work with the columns in this dataset.
- One common example is the use of a dynamic external store to allow MashZone NextGen to work with datasets from distributed stores in the Apama MemoryStore.

With dynamic stores that hold external data, the external system must set a name for the cache manager and define search attributes programmatically when the external system creates the store, using the BigMemory Max API. See BigMemory Max documentation (http://terracotta.org/documentation/) for more information and examples.

Configuration defined in MashZone NextGen for the dynamic store allows RAQL to retrieve this configuration information.

#### 4.10.3.1 Manage Dynamic BigMemory Max Stores for MashZone NextGen Analytics

You must define connections and identify configuration information for BigMemory Max stores that are created by and store data from external systems and then are used as In-Memory Stores in MashZone NextGen Analytics. For in-memory stores that are created dynamically by other systems, MashZone NextGen retrieves configuration and connection information from the Terracotta Management Console (TMC) that manages the host BigMemory Max Server.

You can also define connections to external in-memory stores that are not created dynamically. See Declare BigMemory Max Stores for MashZone NextGen Analytics (page 68) for more information.

For information on the requirements for in-memory stores that act as dynamic external stores for MashZone NextGen Analytics. For instructions on adding and managing external

dynamic store configuration, see Add an External Dynamic In-Memory Store Connection (page 74) and Delete External Dynamic In-Memory Store Connections (page 75).

### 4.10.3.2 Add an External Dynamic In-Memory Store Connection

**Procedure**

1. Verify that the Terracotta Management Console (TMC) that manages the BigMemory Max Server hosting this dynamic store is running and that the store exists.
2. You should also verify that the dynamic store meets minimum requirements for MashZone NextGen.
3. Copy the JAR file containing the classes used as search attributes to extract data from the dataset in this store to <MashZone NextGen-installation/lib.

   See documentation for the external system that created this dynamic store to determine what JAR files are needed. For Apama, see documentation on the MemoryStore.

   The default location for the target folder in MashZone NextGen is <MashZone NextGen-installation>/apache-tomcat/mashzone/WEB-INF/lib. If MashZone NextGen is deployed in a cluster, however, this location may be a separate external folder. For more information, see Setting Up an External MashZone NextGen Configuration Folder (page 111).
4. Restart the MashZone NextGen Server. For instructions, see Start and Stop the MashZone NextGen Server (page 21).
5. Open the MashZone NextGen Admin Console.
6. Click **Server** to expand this section of the Administration menu.
7. Click **BigMemory Max**.
8. Open the **Terracotta Management Server** tab.

   This tab lists connections to any existing external dynamic In-Memory Stores.
9. Click **Register a new Terracotta Management Server**.
10. Enter a unique BigMemory Max data source name for this connection to the dynamic external cache that will act as an In-Memory Store.
11. Enter the domain and port, or IP address and port for the Terracotta Management Server. For example: localhost:9889.
12. Enter the Terracotta Management Server connection name.

    Connection names cannot include periods (.), spaces or other common symbols or punctuation characters.

13. Enter the name of the Cache Manager for this cache. This name is assigned by the external system that created the cache in BigMemory Max.

    Cache Manager names are a best practice for dynamic external stores. If the external system does not assign a cache manager name, BigMemory Max uses a default name which can lead to name collisions and errors.
14. If the TMC requires TLS for connections, change the Security type to TLS.
15. You can enter an user name and a password optionally.
16. Click **Add this external cache** to save this connection.

### 4.10.3.3 Delete External Dynamic In-Memory Store Connections

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Server** to expand this section of the Administration menu.
3. Click **BigMemory Max**.
4. Open the **Terracotta Management Server** tab.

   This tab lists connections to any existing **Terracotta Management Server**.
5. Click **Delete** for the specific connection you want to delete.

### 4.10.3.4 Define permissions for external dynamic In-Memory Store Connections

You can define permissions for external dynamic In-Memory Store connections.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Server** to expand this section of the **Administration** menu.
3. Click **BigMemory Max**.
4. Open the **Terracotta Management Server** tap.

   This tab lists connections to any existing Terracotta Management Server.
5. Click the **Edit permission** icon to configure the permission of an existing Terracotta Management Server.
6. Enter a term in the search field and click **Search**. Clicking on **Search** without any input values fetches all users and groups.

7. Drag a user or user group from the **Search results** field and drop it into the **Principals with permissions** field.

   By default, the owner of the Terracotta Management Server alias is already present in the **Principals with permissions** list. This owner is non editable and cannot be removed from the list.
8. Enable or disable the **View** or **Usage** privileges of a user or user group.
9. Click **OK**.

Your settings are applied.

## 4.11 Manage data sources and drivers

Data sources combine the connection and driver information needed to work with both the MashZone NextGen Repository and with other databases. Data sources can use either JDBC connections or a JNDI connection pool.

See Add a data source (page 76), Edit, test or remove data sources (page 78) and Add or manage JDBC drivers (page 79) for instructions.

### 4.11.1 Add a data source

If you use connection pools to connect to databases, configure JNDI in your application server to enable access to the connection pools as needed.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **JDBC Configuration** tab.
3. If needed, add the JDBC driver for this database to MashZone NextGen. See Add or manage JDBC drivers (page 79) for instructions.
4. Click **Data Sources** to see a list of existing data sources.
5. Click **Add data source** to create a new data source.
6. Enter a data source name for a new data source.

   Data source names may contain ASCII alphabetic characters and numbers **only**. Data source names may **not** contain any punctuation or symbols, such as periods (.), dashes (-) or underscores (_).
7. Select the appropriate driver for this datasource in the **JDBC Driver** drop-down menu.
8. In the **JDBC URL** field, enter either the URL for a JDBC connection or the JNDI name for a connection pool to connect to this data source. Common URL or JNDI forms include:

jdbc:mysql://**hostname**/databasename

For MySQL databases, it is **recommended** that you include the database name in data source URLs. If this information is omitted, testing the data source fails and may also cause errors with access to stored procedures.

jdbc:oracle:**driver-type**@**hostname**:port

jdbc:postgresql://**hostname**:**port**/database-name

jdbc:jtds:sqlserver://**hostname**:**port**;database-name

jdbc:sqlserver://**hostname**:**port**;databaseName=database-name

jdbc:sybase:Tds:**hostname**:port

**java:context-path/jndi-resource-name** or context-path/jndi-resource-name

9. Optionally, enter the username and password to use to connect to this database.
10. Click **Show connection pooling options** to display further options.
11. Optionally, set connection pooling options for this data source:

    **Initial Size** = the initial number of connections to create when the pool for this data source starts up. This defaults to 0.

    **Maximum Active** = the maximum number of connections that can be allocated at one time for this data source. This defaults to 8. Set this to -1 to remove all limits.

    **Maximum Wait** = the maximum number of milliseconds that the pool will wait when no connections are available before failing. Defaults to -1 which is an indefinite wait.

    **Maximum Idle** = the maximum number of connections that can be idle without connections being released for this data source. Defaults to 8. Set this to -1 to prevent any connections being released.

    **Minimum Idle** = the minimum number of idle connections that can exist before new connections are added to the pool for this data source. This defaults to 0, indicating no new connections should be created.

    **Pool Prepared Statement** = set this option to allow prepared statements for the database mashables that use this datasource to be pooled. This is disabled by default.

    The usefullness and effect of pooling prepared statements depends on the type of database for this connection. See documentation for your database for more information or recommendations.

    **Validation Query** = the SQL query that is used to validate connections in the pool for this datasource.

    **Validation Call Timeout** = the number of milliseconds before a connection validation check is considered to have failed, causing the pool to invalidate and discard the connection. If you set this property to a number less than zero, validation calls do not expire, which is the default behavior.

**Time Between Eviction Runs** = the number of milliseconds between tests for idle connections for this datasource. This defaults to -1, which prevents all idle connection testing.

**No of tests per run** = the number of connections to test during any idle test run for this datasource. This defaults to 3.

**Minimum Evictable Idle Time** = the minimum number of milliseconds that a connection can be idle before being tested for eviction. This defaults to 30 minutes (1800000 milliseconds).

For more details on connection pooling properties, see Apache DBCP Documentation.

12. Click **Save changes**.

## 4.11.2 Edit, test or remove data sources

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **JDBC Configuration** tab.
3. Click **Data Sources** to see a list of existing data sources.

   Initially, this lists the data source for the MashZone NextGen Repository and for the Snapshots repository.
4. To edit a data source, click the **Edit** icon on the line for that data source and change properties.

   See Add a data source (page 76) for information on specific data source properties.
5. To test the connection to a data source, click the **Test** icon on the line for that data source.
6. To delete a data source, click the **Delete** icon on the line for that data source.

   Do **not** delete the data source for either the MashZone NextGen Repository or the Snapshots repository.

## 4.11.3 Share data sources

You can share data sources with particular users and user groups so that these have access to **JDBC data sources**.

**Prerequisite**

You have administration privileges.

Regardless of the share, users with administration privilege can access all data sources.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **JDBC Configuration** to expand this section of the **Administration** menu.
3. Click **Data Sources**.
4. Click the **Edit** data source permissions icon of the data source you want to share.
5. Enter a term in the search field and click **Search**. Clicking on **Search** without any input values fetches all users and groups.
6. Drag an user or user group from the **Search result** field and drop it into the **Principals with permissions** field.

   By default, the owner of the data source is already present in the **Principals with permissions** list. This owner is non editable and cannot be removed from the list.
7. Activate or deactivate the Display or Usage privileges of a user or user group.

   A user or user group with **Display** privilege can see the relevant source data in the data feed or dashboard. A user or user group with the **Usage** privilege has access to the relevant alias in the data source operator
8. Click **OK**.

Your changes are applied.

## 4.11.4 Add or manage JDBC drivers

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **JDBC Configuration** tab.
3. Click **JDBC Drivers**.

   A list of existing drivers displays. This initially contains just the driver for the default MashZone NextGen Repository.
4. To remove a driver, click **Delete** on the line for that driver.

   Do **not** delete the driver for the MashZone NextGen Repository.
5. To add a new driver:
   a. Click **Add new JDBC driver**.
   b. Enter a **Name** for a new driver.
   c. Enter the Java Class name for this driver.
   d. Click **Browse** and find the **JAR** file for this driver.

e. Click **Add this JDBC driver**.

## 4.11.5 Migrate JDBC connections

With the MashZone NextGen version 9.10 release the persistence of JDBC drivers and connections have been changed. And only one type of JDBC connections is still available. The current version of MashZone NextGen supports the import of JDBC connections from MashZone legacy, Presto legacy and MashZone NextGen version 9.10.

### 4.11.5.1 Migrate JDBC configuration of Presto to MashZone NextGen

You can import the JDBC configuration of Presto (version 3.9 and 9.9) in MashZone NextGen.

**Procedure**

1. To export all existing JDBC configurations from Presto version 3.9 and 9.9 go to the \prestocli\bin folder of the Presto installation, open a dos command line and call:

   **pAdmin exportDatasource -l http://localhost:8080/mashzone/esd/api -f JDBCConnections_backup.zip -u Administrator -w manage -j**

   The created zip file contains all information about JDBC configurations of Presto, including the related drivers.
2. Copy the **JDBCConnections_backup.zip** file to the **prestocli\bin** folder in your MashZone NextGen installation.
3. Go to **prestocli\bin** folder in your MashZone NextGen installation, open a dos command line and call:

   **pAdmin importDatasource -f JDBCConnections_backup.zip -u Administrator -w manage -o**

All Presto JDBC configurations will be imported into MashZone NextGen.

### 4.11.5.2 Migrate JDBC connections of Presto to MashZone NextGen

You can import the JDBC connections of Presto (version 3.9 and 9.9) in MashZone NextGen.

**Procedure**

This upgrade is relevant for MashZone legacy JDBC connections.

1. Copy all drivers located in the jdbcdrivers folder of your Presto installation into thejdbcdrivers folder of the MashZone NextGen installation and restart the MashZone NextGenMashZone NextGen Server .
2. Check if the MashZone tab exists in the Admin Console of Presto. If not, open the presto.config file located in <Presto installation>\apache-tomcat\webapps\mashzone\WEB-INF\classes\ and set the mashzone.administration.disabled=false flag and restart the Presto server.
3. Export the required connections in the Presto Admin Console (Admin Console -> MashZone -> Database Connections tab. You have to export each connection separately. See Presto online help for details.

   The exported JDBC connections are stored as mzp files, starting with **A_DATABASE...**, in the **importexport** folder of your Presto installation.
4. Copy all database related mzp files in the importexport folder of your Presto installation to the dbconnections subfolder of the importexport folder in your MashZone NextGen installation.
5. Start the MashZone NextGen server if not already done. Then go to the runtool folder (located under Presto\mashzone\tools), open a dos command line and call:

   **migrationtool -user Administrator -password manage -folder dbconnections**

All JDBC connections from the **dbconnections** folder will be imported into MashZone NextGen.

In the MashZone NextGen Admin Console the JDBC connections are separated in two parts, the driver part and the data source part. You can find all JDBC related items in the **JDBC Configuration** tab of the Admin Console.

If you need to upgrade a connection using a JDBC driver that consists of multiple JAR files, you will have to create a new driver JAR which bundles all the individual files into one single file. After that, you will have to copy the newly created JAR file to the MashZone NextGen installation.

### 4.11.5.3 Migrate JDBC configuration of MashZone NextGen 9.10

You can import the JDBC configurations of MashZone NextGen version 9.10 in the current MashZone NextGen version.

**Procedure**

1. To export all existing JDBC configurations from MashZone NextGen 9.10 go to the \prestocli\bin folder of the MashZone NextGen 9.10 installation, open a dos command line and call:

   **pAdmin exportDatasource -l http://localhost:8080/mashzone/esd/api -f JDBCConnections_backup.zip -u Administrator -w manage -j**

   The created zip file contains all information about JDBC configurations of MashZone NextGen 9.10, including the related drivers.
2. Copy the **JDBCConnections_backup.zip** file to the **prestocli\bin** folder in your current MashZone NextGen installation.
3. Go to **prestocli\bin** folder in your current MashZone NextGen installation, open a dos command line and call:

   **pAdmin importDatasource -f JDBCConnections_backup.zip -u Administrator -w manage -o**

All MashZone NextGen 9.10 JDBC configurations will be imported into the current MashZone NextGen version.

#### 4.11.5.4 Migrate JDBC connections of MashZone legacy to MashZone NextGen

You can import the JDBC connections of MashZone legacy (versions 9.5 to 9.12) in MashZone NextGen.

See the **MashZone NextGen Migration Guide** for details.

## 4.12 Manage geographical map ressources

### 4.12.1 Manage geoJSON files

GeoJSON is an open-standard format designed for representing simple geographical features, along with their non-spatial attributes. It is based on JSON, the JavaScript Object Notation. You can upload GeoJSON files to MashZone NextGen to use in a custom map style.

All geoJSON files are stored in the following directory:

**<MashZone NextGen installation>\apache-tomcat\webapps\mashzone\hub\dashboard\assets\geomap\{tenant}\vectormaps**

To support multi-tenancy, you must replace {tenant} in the path above with the corresponding tenant name. If there are no tenants, **default** is used in place of {tenant} (tenant name). If a tenant does not have a folder, the **shared** folder is used.

By default, MashZone NextGen is shipped with a few default geoJSON files in the following path:

<MashZone NextGen installation>\apache-tomcat\webapps\mashzone\hub\dashboard\assets\geomap\default\vectormaps

The map styles are available in the **Vector map** widget and can be selected there by file name in the **Template** drop-down menu.

The following map style templates are included:

- world-countries-by-name (default)
- world-countries-by-iso3
- world-continents-by-name
- us-states-by-name

If you add you own map style templates, we recommend that all defined regions (features) are polygons. All non-polygon regions (features) are ignored. The **ID** attribute is the default identifier for each region (feature). The data assigned to the map widget must contain a column with values matching the **ID** attribute. Therefore, any polygon without an **ID** attribute is also ignored. If no data is assigned, you can use the vector map to select a region and publish such a selection using the region ID.

### 4.12.2 Manage tile server configuration files

TMS (Tile Map Service) is a protocol for serving maps as tiles, which involves splitting the map up into a pyramid of images at multiple zoom levels. You can upload your own tile server configuration files to MashZone NextGen.

All tile server configuration JSON files are in the following directory:

**<MashZone NextGen installation>\apache-tomcat\webapps\mashzone\hub\dashboard\assets\geomap\{tenant}\tileservers**

To support multi-tenancy, you must replace {tenant} in the path above by the corresponding tenant name. If there are no tenants, **default** is used in place of {tenant} (tenant name). If a tenant does not have a folder, the **shared** folder is used. By default, MashZone NextGen is shipped with a default tile server configuration. It enables you to use the Open Street Map tile servers, which have been included as a point of reference.

The maps are available in the **Map with markers** widget and can be selected there in the **Base map** drop-down menu.

You can also add a **LeafletJS** compatible tile server configuration to host an Open Street Map data-based tile server yourself. You must be aware of all terms and licensing conditions that may apply when using a third-party tile server, including servers hosted by Open Street Map.

**Example**

The following configuration example points to Open Street Map tile servers hosted by Stamen.

```
{
    "label": "Stamen (watercolor)",
    "url":"http://{s}.tile.stamen.com/watercolor/{z}/{x}/{y}.jpg",
    "attribution":"Map tiles by <a href=\"http://stamen.com/\"
target=\"_new\">Stamen Design</a> Data by <a
href=\"http://openstreetmap.org/\" target=\"_new\">OpenStreetMap</a>",
    "subdomains": ["a","b","c","d"]
}
```

## 4.13 Tune memory/caching for MashZone NextGen

In large installations with many users and/or many calculations, it makes sense to increase the Java heap memory, the sizes of the feed result caches and the number of calculation threads using the following techniques:

- Tune MashZone NextGen memory and cache configuration manually (page 84).
- Manual tuning gives you control to balance memory requirements for MashZone NextGen, but does require manual updates to several configuration files.

### 4.13.1 Tune MashZone Memory and Cache Configuration Manually

To manually update memory and cache configuration

- Update Cache Memory Settings (page 85)
- Update MashZone ThreadSize Properties (page 85).
- Then restart the MashZone NextGen Server to apply this change. See Start and Stop the MashZone NextGen Server (page 21) for instructions.

### 4.13.2 Update Cache Memory Settings

- In the text editor of your choice, open the **ehcache.xml** file in the web-apps-home/mashzone/WEB-INF/classes folder.
- Update the **maxBytesLocalHeap** value on the **<cache>** elements with the following names:
- **RESULT_FEED_BASE**
- **RESULT_FEED_TOP**
- **RESULT_FEED_DEBUG**
- Save your changes

### 4.13.3 Update MashZone ThreadSize Properties

- In the text editor of your choice, open the **mashzone.properties** file in the web-apps-home/mashzone/WEB-INF folder.
- Update the following properties:
- **calculation.threadpool.coresize**
- **calculation.threadpool.maxsize**
- Save your changes

## 4.14 Disable automatic masking in CSV files

You can disable the automatic masking when you export data as a CSV file.

By default, all values beginning with =, +, -, or @ are automatically masked by a preceding single quotation mark before exported as CSV file. For details on exporting data as a CSV file, see the chapter Save widget data as a CSV file.

**Procedure**

1. Open the **mashzone.properties** file in the following directory.
   <MashZone NextGen installation>/apache-tomcat/webapps/mashzone/WEB-INF/
2. Set the **allow.secure.csv.export parameter** to false as follows.
   allow.secure.csv.export=false

The automatic masking when exporting CSV files is disabled.

# 5 MashZone NextGen Server Administration

## 5.1 Manage files for MashZone NextGen dashboards and data feeds

MashZone NextGen uploads and hosts files for dashboards and data feeds that are not accessible via HTTP (spreadsheets, CSV or XML files). These files are saved and managed in the MashZone NextGen Repository to ensure better management of resources and easier deployment or migration across different environments and versions.

MashZone NextGen administrators may need to manually add files to MashZone NextGen to provide data files or resources for dashboards or data feeds.

Common management tasks for files include:

- Add external resources as MashZone NextGen files (page 86)
- Find MashZone NextGen Files (page 87)
- Update or delete MashZone NextGen files (page 87)
- Share MashZone NextGen resource files (page 88)

### 5.1.1 Add external resources as MashZone NextGen files

You can add external resources to MashZone NextGen to make them easily accessible in dashboards and data feeds.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **Platform Features** section and click **File Resources**.
3. Click **Upload New File**.
4. Click **File**, find and select the file you want to upload and click **Open**.

   The location and file name are filled in and a new set of fields open to upload another file.
5. If needed, add to the path or change the file name.

   The default file name is /**file-name**. If you accept the default, the URL to access this file becomes http://app-server:port/mashzone/files/file-name.

   You can organize files into 'pseudo folders' by adding to the path, using a slash **/** as the separator. For example, a file name of **/images/reports.png** has a URL of **http://app-server:port/mashzone/files/images/reports.png** and can be found in file search (along with any other files in the 'images folder') by searching for **images** as the file name.

You can also upload files that are normally loaded automatically, such as thumbnails. Simply specify the standard path.

6. Repeat the steps, as needed, to find and name any other files you want to upload.

The files are added to the MashZone NextGen Repository and are now available via a MashZone NextGen URL.

### 5.1.2 Find MashZone NextGen files

You can search for specific resource files.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **Platform Features** section and click **File Resources**.
3. Enter either:

   Part of the file name(s).

   Part of the path to the file(s).
4. Click **Search**.

The search result is displayed. The file search results are always sorted by path and file name.

### 5.1.3 Update or delete MashZone NextGen files

Although rare, you may occasionally need to update or even delete files from MashZone NextGen.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **Platform Features** section and click **File Resources**.
3. Find the specific file you need to update or delete. See Find MashZone NextGen files (page 87) for techniques.
4. To upload an updated file:
   a. Click **Edit** on the line for that file.
   b. Click **File** and find the updated file you want to replace the existing file in MashZone NextGen.
   c. Click **Upload this file**.

5. To delete a file, click **Delete** on the line for that file.

### 5.1.4 Share MashZone NextGen resource files

You can share resource files with particular users and user groups so that these have access to the files. MashZone NextGen users can use these files as external resources in their dashboards and data feeds.

Regardless of the share, users with administration privilege can access all resource files. By default, all other users only have access to images (such as PNG, JPG, or GIF).

**Prerequisite**

You have administration privileges.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Expand the **Platform Features** section and click **File Resources**.
3. Click **Edit file permission** to configure the file permission for an existing file resource.
4. Enter a term in the search field and click **Search**. Clicking on **Search** without any input values fetches all users and groups.
5. Drag a user or user group from the **Search results** field and drop it into the **Principals with permissions** field.

   By default, the owner of the resource file is already present in the **Principals with permissions** list. This owner is non editable and cannot be removed from the list.
6. Enable or disable the **Usage** privileges of a user or user group.

   A user or user group with **Usage** privilege has access to the corresponding resource file and can use it in data feeds or dashboards.
7. Click **OK**.

Your settings are applied.

## 5.2 Manage resource directories

Resource directories hold file-based data sources, such as Excel spreadsheets, CSV or XML files.

The resource aliases can be used by the data source operators to read local files.

### 5.2.1 Create resource directory

To work with data sources in MashZone NextGen Feed Editor that are file-based, such as Excel spreadsheets, CSV files or XML files, you must store the files in a **resource directory** that the Integrated MashZone NextGen Server knows. This can be the default resource directory:

MashZoneNG-install/mashzone/data/resources

Or it can be any subdirectory of the default.

You can also use resource directories to control access to data source files to specific users or groups.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **File resource** -> **File resource** to open the resource alias page.
3. Click **Create**.
4. Give the directory an alias name of your choice in the **Resource directory** input box.
   You cannot modify the alias name later.
5. Enter the Path of the new resource directory.
6. Click **Add resource**.

The new resource directory is created and is displayed in the list with the specified alias.

### 5.2.2 Change resource directory

You can adapt the path of already existing resource directories.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **File resource** -> **File resource** to open the resource alias page.
3. Click the **Edit resource alias** icon of the resource you want to edit.
4. Enter the **Path** of the resource directory.
5. Click **Save resources**.

Your changes are applied.

### 5.2.3 Delete resource directory

You can delete existing resource directories.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click File resource -> File resource to open the resource alias page.
3. Click the **Delete resource alias** icon of the resource you want to delete.
4. Click **Yes**.

The directory selected is deleted from the list.

### 5.2.4 Share resource directory

You can share resource directories with particular users and user groups so that these have access to the directory content.

Regardless of the share, users with administration privilege can access all resource directories.

**Prerequisite**

You have administration privileges.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **File alias** -> **File alias** in the **Administration** menu.
3. Click the **Edit resource permissions** icon of the resource you want to share.
4. Enter a term in the search field and click **Search**. Clicking on **Search** without any input values fetches all users and groups.
5. Drag an user or user group from the **Search result** field and drop it into the **Principals with permissions** field.

   By default, the owner of the resource directory is already present in the **Principals with permissions** list. This owner is non editable and cannot be removed from the list.
6. Click **OK**.

Your settings are applied.

## 5.3 Manage URL aliases

You can manage your URL aliases in the **Admin console**.

It is always recommended using a URL alias to shorten the link used in e.g. dashboards and data feeds. You have to enter the path where the data are stored only, and not the complete URL. The resource aliases can be used by the data source operators to read local files.

### 5.3.1 Create URL alias

You can create URL aliases to shorten a link used in, for example, dashboards and data feeds.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **URL aliases** -> **URL aliases** to open the **URL alias** page.
3. Click **Create**.
4. Give the URL an alias name of your choice in the **Alias** input box.

   You cannot modify the alias name later.
5. Enter the URL in the corresponding input box.
6. Enable the **Use basic authentication** option if an authentication is require to use the URL.

   Enter the user name and password associated with the user name in the corresponding input boxes.
7. Click **Add alias**.

The new URL alias is created and is displayed in the **URL alias** list.

### 5.3.2 Change URL alias

You can adapt the URL and the authentication credentials of already existing URL aliases.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **URL aliases** -> **URL aliases** to open the **URL alias** page.
3. Click the **Edit URL alias** icon of the URL alias you want to edit.
4. Make your changes.
5. Click **Save**.

Your changes are applied.

### 5.3.3 Delete URL alias

You can delete existing URL aliases.

1. Open the MashZone NextGen Admin Console.
2. Click **URL aliases** -> **URL aliases** to open the **URL alias** page.

3. Click the **Delete URL alias** icon of the URL alias you want to delete.
4. Click **Yes**.

The URL alias selected is deleted from the list.

### 5.3.4 Share URL alias

You can share URL aliases with particular users and user groups so that these have access to the directory content.

**Prerequisites**

You have administration privileges.

Regardless of the share, users with administration privilege can access all URL aliases.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **URL aliases** -> **URL aliases** to open the **URL alias** page.
3. Click the **Edit URL alias permissions** icon of the alias you want to share.
4. Enter a term in the search field and click **Search**. Clicking on **Search** without any input values fetches all users and groups.
5. Drag an user or user group from the **Search result** field and drop it into the **Principals with permissions** field.

   **Note:** By default, the owner of the URL alias is already present in the **Principals with permissions** list. This owner is non editable and cannot be removed from the list.
6. Activate or deactivate the **Display** or **Usage** privileges of a user or user group.

   A user or user group with **Display** privilege can see the relevant source data in the data feed or dashboard. A user or user group with the **Usage** privilege has access to the relevant alias in the data source operator.
7. Click **OK**.

Your changes are applied.

## 5.4 Deploying MashZone NextGen instances, clusters, or artifacts

Deploying MashZone NextGen to new hosts or new environments typically involves Deploying the Core Widgets (page 93), shown below, and optionally Deploying MashZone NextGen Artifacts and Other Metadata (page 93).

### 5.4.1 Deploying the core widgets

The core widgets include the MashZone NextGen Server, the MashZone NextGen Repository, which is typically installed in a database other than the default Derby database, and the MashZone NextGen Analytics In-Memory Stores and MashZone NextGen caches.

In earlier releases, the MashZone NextGen Hub and AppDepot were deployed in a separate web application from the MashZone NextGen web application. Effective in 3.2, all the core widgets are deployed in the single web-apps-home/presto web application.

For individual MashZone NextGen servers, you typically do a default installation (see Installing Software AG Products). You may also move the MashZone NextGen Repository to a database of your choice. See Move the MashZone NextGen repository to a robust database solution (page 24) for instructions.

You can leave the MashZone NextGen Analytics In-Memory Stores and MashZone NextGen caches in local memory for a single MashZone NextGen server. This uses the default client installation of BigMemory Max. If additional memory or reliability is required, you can also deploy BigMemory Max as an add-on in a separate host or cluster. See Working with MashZone NextGen Analytics In-Memory Stores for more information and links.

To deploy multiple unclustered servers, see Deploying Multiple MashZone NextGen Servers in One Host (page 104). To deploy MashZone NextGen servers in clusters, see Clustering MashZone NextGen Servers (page 107) for requirements and links.

### 5.4.2 Deploying artifacts and other metadata using the command line

You deploy specific artifacts and metadata from a source MashZone NextGen Server to a target MashZone NextGen Server using the export and import commands.

**Important:** You **cannot** use export and import commands when the MashZone NextGen version for the source and target MashZone NextGen Servers are different:

- For major upgrades, use the MashZone NextGen migration utility instead.

  For details on using the migration utility, please refer the MashZone NextGen Migration Guide.
- For minor upgrades, please contact Technical Support or your Software AG representative.

In addition to the basic metadata for an artifact, a successful deployment must include related metadata, related files, extensions the artifact may use and any other artifacts that the artifact depends on.

The export and import commands automate deployment for most of this data, with some specific limitations that require manual deployment steps.

**Procedure**

1. Export the specific artifacts that you want to deploy to another MashZone NextGen Server and any artifacts that they may use.

   See the following topic for instructions using these MashZone NextGen export commands:

   Export users, groups, and role assignments (page 95)

   Export dashboards (page 97)

   Export data feeds (page 98)

   Export aliases (page 99)

2. Copy the files for any extensions used by the exported artifacts from the MashZoneNG-config folder for the source MashZone NextGen Server to the MashZoneNG-config folder for the target MashZone NextGen Server.

   **Note:** The MashZoneNG-config folder may be an external configuration folder outside of the source and target MashZone NextGen Servers or it may be in the default locations. See Setting Up an External MashZone NextGen Configuration Folder (page 111) for more information on MashZoneNG-config locations.

3. Define datasources in the Admin Console for the target MashZone NextGen Server with matching names and JDBC drivers to the datasources in the source MashZone NextGenMashZone NextGen Server.

   See Manage data sources and drivers (page 76) for instructions.

4. Use the export files created earlier to import dashboards and data feeds, users, groups and user group assignments from the source MashZone NextGen Server.

   See the following topics for information on using these commands:

   Import dashboards (page 102)

   Import data feeds (page 103)

   Import users, user metadata and groups (page 100)

   Import aliases (page 104)

**Table 1. Known Export/Import Limitations**

| | Exported | Not Exported |
|---|---|---|
| Related Metatdata/ User Metadata | ▪ Users, groups and user group assignments if this data is tracked in the default | ▪ Datasources and their JDBC drivers that are used by dashboards and data feeds. |

| | | |
|---|---|---|
| | MashZone NextGen User Repository and not in your LDAP Directory. | ▪ Datasources **must** be added to the target MashZone NextGen Server before you import any artifacts that use them or the import will fail. |
| MashZone NextGen Server Configuration | | Configuration for the MashZone NextGen Server. |

### 5.4.2.1 Export users, groups, and role assignments

You can export all users, groups, and role assignments from the MashZone NextGen Repository to an export file. You can then import this file to another MashZone NextGen Repository.

**Important:** If you have configured MashZone NextGen to work with your LDAP Directory, this command **only** exports MashZone NextGen User Attributes. Data for users, user groups and user group assignments resides in LDAP.

**Procedure**

1. If it is not running, start the MashZone NextGen Server for the MashZone NextGen Repository with the user groups that you wish to export. See Start and Stop the MashZone NextGen Server (page 21) for instructions.
2. Open a command window and move to the **<MashZone NextGen installation>/prestocli/bin** folder.
3. Enter this command:

   **padmin exportUsersAndGroups -f output-file [-l prestoURL] -u username -w password [-v]**

   -f output-file: is the path and name of the export file to hold the metadata.

   -l prestoUrl: is optional. Use this if the MashZone NextGen Server is remote or if it is not running in Tomcat on the default Tomcat port. If you omit this option, this defaults to **http://localhost:8080/mashzone/**.

   -u username: is the MashZone NextGen username to log in with. This account must have MashZone NextGen administrator permissions.

   -w password: is the MashZone NextGen password to log in with.

   **-v**: is an optional flag to turn on verbose logging.

All messages and errors from the export process are sent to the command window (stdout). Once the export command completes successfully, you can use the output file to import the data to another MashZone NextGen Repository.

### 5.4.2.2 Export role assignments for users and groups

You can export all role assignments for users and groups from the MashZone NextGen Repository to an export file. You can then import this file to another MashZone NextGen Repository.

**Important:** If you have configured MashZone NextGen to work with your LDAP Directory, this command **only** exports MashZone NextGen User Attributes. Data for users, user groups and user group assignments resides in LDAP.

**Procedure**

1. If it is not running, start the MashZone NextGen Server for the MashZone NextGen Repository with the user groups that you wish to export. See Start and Stop the MashZone NextGen Server (page 21) for instructions.
2. Open a command window and move to the **<MashZone NextGen installation>/prestocli/bin** folder.
3. Enter this command:

   **padmin exportRoleAssignments -f output-file [-l prestoURL] -u username -w password [-v]**

   -f output-file: is the path and name of the export file to hold the metadata.

   -l prestoUrl: is optional. Use this if the MashZone NextGen Server is remote or if it is not running in Tomcat on the default Tomcat port. If you omit this option, this defaults to **http://localhost:8080/mashzone/**.

   -u username: is the MashZone NextGen username to log in with. This account must have MashZone NextGen administrator permissions.

   -w password: is the MashZone NextGen password to log in with.

   **-v**: is an optional flag to turn on verbose logging.

All messages and errors from the export process are sent to the command window (stdout). Once the export command completes successfully, you can use the output file to import the data to another MashZone NextGen Repository.

### 5.4.2.3 Export dashboards

You can export your MashZone NextGen dashboards.

You can use the export zip file to create a backup or to import your dashboards into another MashZone NextGen installation.

**Procedure**

1. Open a command window and move to the **MashZoneNG-install/prestocli/bin** folder.
2. Enter this command:

   padmin exportDashboard -i **identify** [-f **output-file]** [-l **prestoURL]** -u **username** -w **password** [-v] [-o]

   **-i** identify: Mandatory dashboard identifier. It can be "id=", "name=" or "all", enclosed in quots.

   Examples

   -i "name=dashboardname": If there are multiple dashboards with the same name only the first dashboard found will be exported.

   -i "id=43243244434432": The dashboard ID (GUID) is unique in the MashZone NextGen system.

   -i "all": Exports all dashboards for that user.

   **-f** output-file: Optional path and name for the export. If omitted, an output zip file is created in the folder in which this command is executed:

   Single export with option -i "id=3456" or "name=name" create a new file with name "name_guid.zip"

   Multiple export with option -i "all" create a new file dashboard-export-timestamp.zip

   **-l** prestoUrl: is optional. Use this if the MashZone NextGen Server is remote or if it is not running in Tomcat on the default Tomcat port. If you omit this option, this defaults to **http://localhost:8080/mashzone/**.

   **-u** username: MashZone NextGen user name to log in with. This account must have MashZone NextGen administrator permissions.

   **-w** password: is the MashZone NextGen password to log in with.

   **-v**: is an optional flag to activate verbose logging.

   **-o**: Optional flag to overwrite an existing export file.

Once the export command completes successfully, you can use the output file to import dashboards into MashZone NextGen.

Permissions for each dashboard were automatically stored in the zip file. If no permissions are assigned to the dashboard, the permission file saved is empty.

The zip file also includes information about the dashboard creator.

### 5.4.2.4 Export data feeds

You can export your MashZone NextGen data feeds.

Export creates an export file that you can use to import data feeds to MashZone NextGen.

**Procedure**

1. Open a command window and move to the **<MashZone NextGen installation>/prestocli/bin** folder.
2. Enter this command:

   padmin exportFeed -i identify [-f output-file] [-l prestoURL] -u username -w password [-v] [-o]

   Examples

   -i identify: mandatory data feed identifier. It can be "id=", "name=" or "all", enclosed in quots.

   -i "name=feedname": If there are multiple data feeds with the same name then only the first founded data feed will be exported.

   -i "id=43243244434432": The data feed id (Guid) is unique in the MashZone NextGen system.

   -i "all": Export of all data feeds for that user.

   **-f** output-file: an optional path and name for the export file to put data feeds. If omitted, this generates an output ZIP file in the folder where this command is executed:

   Single export with option -i "id=3456" or "name=name" create a new file with name "name_guid.zip"

   Multiple export with option -i "all" create a new file datafeed-export-timestamp.zip

   This file must not already exist, unless you also use the **-o** option.

   **-l** prestoUrl: is optional. Use this if the MashZone NextGen Server is remote or if it is not running in Tomcat on the default Tomcat port. If you omit this option, this defaults to **http://localhost:8080/mashzone/**.

   **-u username: is the MashZone NextGen username to log in with. This account must** have MashZone NextGen administrator permissions.

   **-w** password: is the MashZone NextGen password to log in with.

   **-v**: is an optional flag to turn on verbose logging.

   **-o**: an optional flag to overwrite an existing export file. If you omit this option, the output file must not already exist.

Once the export command completes successfully, you can use the output file to import data feeds to MashZone NextGen.

Permissions for each data feed were automatically stored into the ZIP file. If there are not any permissions assigned to the data feed an empty permission file is stored.

There is also an information about the data feed creator stored in the zip file.

### 5.4.2.5 Export aliases

You can export your MashZone NextGen aliases.

Export creates an export file that you can use to import aliases to MashZone NextGen.

**Procedure**

1. Open a command window and move to the **<MashZone NextGen installation>/prestocli/bin** folder.
2. Enter this command:

   padmin exportAlias -i identify [-f output-file] [-l prestoURL] -u username -w password [-o] [-j]

   **-i** identifier: mandatory alias identifier. Supported identifiers are: type, name or all Supported types: PPM, FILE, DATABASE, EVENT, URL, BIGMEMORY_CONNECTION

   Examples

   - i "type=PPM": Aliases with type **PPM**' are exported.
   - i "name=UMG_EN": Aliases with name **UMG_EN** are exported.
   - i "all": Export of all aliases for that user.

   **-f** output-file: an optional path and name for the export file to put data feeds. If omitted, this generates an output ZIP file in the folder where this command is executed:

   Export creates a new file alias-export-<timestamp>.zip

   This file must not already exist, unless you also use the -o option.

   **-l** prestoUrl: is optional. Use this if the MashZone NextGen Server is remote or if it is not running in Tomcat on the default Tomcat port. If you omit this option, this defaults to http://localhost:8080/mashzone/.

   **-u** username: is the MashZone NextGen username to log in with. This account must have MashZone NextGen administrator permissions.

   **-w** password: is the MashZone NextGen password to log in with.

   **-j** is an optional flag to turn on inclusion jdbc driver JAR files in export.

   **-o** an optional flag to overwrite an existing export file. If you omit this option, the output file must not already exist.

Once the export command completes successfully, you can use the output file to import aliases to MashZone NextGen.

Permissions for each alias were automatically stored into the ZIP file. If there are not any permissions assigned to the alias an empty permission file is stored.

There is also an information about the alias creator stored in the ZIP file.

### 5.4.2.6 Import users, groups, and role assignments

You must have a users export file to import. For details, see the chapter Export users, groups, and role assignments (page 95).

This command is available only in MashZone NextGen 3.2 or later.

A user export file contains MashZone NextGen User Attributes. It can also contain users, user groups, and role assignments if you are using the default MashZone NextGen User Repository rather than an LDAP Directory.

**Procedure**

1. If it is not started, start the MashZone NextGen Server for the MashZone NextGen Repository where you wish to import data. See Start and Stop the MashZone NextGen Server (page 21) for instructions.
2. Open a command window and move to the **<MashZone NextGen installation>/prestocli/bin** folder.
3. Enter this command:

   padmin importUsersAndGroups -f input-file [-l prestoURL] -u username -w password [-c] [-o] [-v]

   -**f** input-file: is the path and name of the export file to import data from.

   -**l** prestoUrl: is optional. Use this if the MashZone NextGen Server is remote or is not running in Tomcat on the default Tomcat port. If you omit this option, this defaults to **http://localhost:8080/mashzone/**.

   **-u** username: is the MashZone NextGen username to log in with. This account must have MashZone NextGen administrator permissions.

   -**w** password: is the MashZone NextGen password to log in with.

   **-c**: is an optional flag to allow the import process to continue if errors occur during the import. By default, any import errors stop all further processing.

   **-o**: is an optional flag to allow import information for a MashZone NextGen global attribute to overwrite an existing user, group, user group assignments or MashZone NextGen User Attribute with the same ID.

**-v**: is an optional flag to turn on verbose logging.

Messages and errors from the import process are sent to the console window (stdout). Once the import is successfully finished, you may confirm that the appropriate data has been imported in MashZone NextGen.

### 5.4.2.7 Import role assignements for users and groups

You must have a users export file to import. For details, see the chapter Export role assignments for users and groups (page 96).

This command is available only in MashZone NextGen 3.2 or later.

A user export file contains MashZone NextGen User Attributes. It may also contain users, user groups and user group assignments if you are using the default MashZone NextGen User Repository rather than an LDAP Directory.

**Procedure**

1. If it is not started, start the MashZone NextGen Server for the MashZone NextGen Repository where you wish to import data. See Start and Stop the MashZone NextGen Server (page 21) for instructions.
2. Open a command window and move to the **<MashZone NextGen installation>/prestocli/bin** folder.
3. Enter this command:

   padmin importRoleAssignments -f input-file [-l prestoURL] -u username -w password [-c] [-o] [-v]

   -**f** input-file: is the path and name of the export file to import data from.

   -**l** prestoUrl: is optional. Use this if the MashZone NextGen Server is remote or is not running in Tomcat on the default Tomcat port. If you omit this option, this defaults to **http://localhost:8080/mashzone/**.

   **-u** username: is the MashZone NextGen username to log in with. This account must have MashZone NextGen administrator permissions.

   -**w** password: is the MashZone NextGen password to log in with.

   **-c**: is an optional flag to allow the import process to continue if errors occur during the import. By default, any import errors stop all further processing.

   **-o**: is an optional flag to allow import information for a MashZone NextGen global attribute to overwrite an existing user, group, user group assignments or MashZone NextGen User Attribute with the same ID.

   **-v**: is an optional flag to turn on verbose logging.

Messages and errors from the import process are sent to the console window (stdout). Once the import is successfully finished, you may confirm that the appropriate data has been imported in MashZone NextGen.

### 5.4.2.8 Import dashboards

You can import dashboards in MashZone NextGen.

The dashboards are saved in a ZIP file containing the dashboard definition, resource policy, and dashboard permissions, etc. If you import a dashboard including permissions, the creator of the dashboard can view and edit the dashboard. The importer of a dashboard automatically becomes the creator of the dashboard if the dashboard is imported without permissions.

**Procedure**

1. Open a command window and move to the <MashZone NextGen installation>/prestocli/bin folder.
2. Enter the following command:

   padmin importDashboard [-l prestoURL] -f input-file -p importPermissions -u username -w password [-v] [-o] [-c]

   **-f** input-file: Path and name of the import ZIP file.

   **-p** importPermissions: Imports the resource policy and permissions saved in the import ZIP file.

   The importer of a dashboard automatically becomes the creator of the dashboard if the dashboard is imported without permissions. And only administrators can see and work with the dashboards imported.

   **-o**: Optional. Allows overwriting an existing dashboard in MashZone NextGen Dashboard.

   **-l** prestoUrl: Optional. Use this if the MashZone NextGen Server is remote or if it is not running in Tomcat on the default Tomcat port. If you omit this option, this results in http://localhost:8080/mashzone/esd/api.

   **-u** username: Is the MashZone NextGen user name to log in with. This account must have MashZone NextGen administrator permissions.

   **-w** password: Is the MashZone NextGen password to log in with.

   **-v**: Optional. Enables the verbose logging.

   **-c**: Optional. Checks whether the selected file can be imported successfully.

Once the import command completes successfully, you can use the imported dashboards in MashZone NextGen.

If you have imported dashboards from Presto 3.9 into MashZone NextGen, save the imported dashboards in edit mode of the widget before you display them in view mode. Otherwise, an error message is displayed.

### 5.4.2.9 Import data feeds

You can import data feeds to MashZone NextGen.

The data feeds are saved in a ZIP file that contains among other things the data feed definition, resource policy and data feed permissions. If you import a data feed including the permissions then the creator of the data feed can view and edit the data feed. Importing data feeds without the relevant permissions makes the importer automatically to the creator of these data feeds.

**Procedure**

1. Open a command window and move to the **<MashZone NextGen installation>/prestocli/bin** folder.
2. Enter this command:

   padmin importFeed [-l prestoURL] -f input-file -p importPermissions -u username -w password [-v] [-o]

   **-f** input-file: path and name for the import ZIP file.

   **-p** importPermissions Imports the resource policy and permissions saved in the import ZIP file.

   If you import data feeds without permissions makes the importer automatically to the creator of these data feeds and the data feeds has no explicit permissions which means that only administrators can see and work with the data feeds .

   **-o**: is optional. Allows to overwrite an existing data feeds.

   **-l** prestoUrl: is optional. Use this if the MashZone NextGen Server is remote or if it is not running in Tomcat on the default Tomcat port. If you omit this option, this defaults to http://localhost:8080/mashzone/esd/api.

   **-u** username: is the MashZone NextGen username to log in with. This account must have MashZone NextGen administrator permissions.

   **-w** password: is the MashZone NextGen password to log in with.

   **-v**: is an optional flag to turn on verbose logging.

Once the import command completes successfully, you can use the imported data feeds in MashZone NextGen feed editor.

## 5.4.2.10 Import aliases

You can import aliases to MashZone NextGen.

The aliases are saved in a ZIP file that contains among other things the alias definitions and permissions.

**Procedure**

1. Open a command window and move to the **<MashZone NextGen installation>/prestocli/bin** folder.
2. Enter this command:

   padmin importAlias [-l prestoURL] -f input-file -p importPermissions -u username -w password [-v] [-o]

   **-f** input-file: path and name for the import ZIP file.

   **-p** importPermissions: Imports the permissions saved in the import ZIP file.

   **-o**: is optional. Allows to overwrite an existing alias.

   **-l** prestoUrl: is optional. Use this if the MashZone NextGen Server is remote or if it is not running in Tomcat on the default Tomcat port. If you omit this option, this defaults to http://localhost:8080/mashzone/esd/api.

   **-u** username: is the MashZone NextGen username to log in with. This account must have MashZone NextGen administrator permissions.

   **-w** password: is the MashZone NextGen password to log in with.

   **-v**: is an optional flag to turn on verbose logging.

Once the import command completes successfully, you can use the imported aliases in MashZone NextGen.

## 5.4.2.11 Deploying multiple MashZone NextGen servers in one host

You can deploy several different, independent MashZone NextGen Servers on a single host. Each MashZone NextGen Server must be hosted in its own application server and have its own MashZone NextGen Repository.

To host multiple, independent servers, simply install each being sure to change the ports assigned to each MashZone NextGen Server, MashZone NextGen Repository and the administration port for Tomcat.

**Note:** You can also create clusters of MashZone NextGen Servers to provide load balancing. See Clustering MashZone NextGen Servers (page 107) for information.

## 5.4.3 Deploying artifacts using the Admin Console

You can export and import dashboards, data feeds, and aliases using the MashZone NextGen Admin Console.

### 5.4.3.1 Export dashboards, data feeds, and aliases

You can export dashboards, data feeds, and aliases using the MashZone NextGen Admin Console, for example, to share your dashboards with other users.

You can export all existing artifacts of a type at once or individual artifacts. The artifacts are exported including the corresponding permissions. The exported artifacts are downloaded and stored in ZIP files in the client system.

**Prerequisites**

You have administration privileges.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Import/Export** -> **Import/Export** in the **Administration** menu.
3. Select the artifact type you want to export from the **Type** drop-down menu.
4. To additionally export the relevant JDBC drivers for the aliases, enable the **Include JDBC drivers** option. The option is only available for aliases.
5. Click **Export all** to export all available artifacts of the selected type.

   All artifacts are stored in one ZIP file.
6. To export a specific artifact, you can specify the corresponding name, or for dashboards and data feeds the GUID. For aliases, you can specify the alias type instead of the GUID, for example, aliases of type EVENT or PPM.
   a. Enable the **Name** option if you want to identify the artifact by name.

      Note that there may be multiple dashboards or data feeds with the same name. A name of an artifact is not unique in MashZone NextGen. All dashboards or data feeds with the same name are exported.
   b. Enable the **GUID** option if you want to identify the artifact by GUID.

      A GUID is unique in MashZone NextGen.

c. Enable the **Type** option if you want to identify the alias by type.

   All aliases of the same type, such as EVENT or PPM, are exported.

d. Enter the name, GUID, or type in the **Identifier** input box.

   An automatic completion function helps you to find the desired artifact and shows the relevant artifacts.

7. Click **Export**.

The selected artifacts are exported.

### 5.4.3.2 Import dashboards, data feeds, and aliases

You can import dashboards, data feeds, and aliases using the MashZone NextGen Admin Console.

The artifacts are stored in a ZIP file containing the artifact definitions and permissions, etc. If you import an artifact including permissions, the creator of the artifact can view and edit the artifact. The importer of an artifact automatically becomes the creator of the artifact if the artifact is imported without permissions.

#### Prerequisites

You have administration privileges.

#### Procedure

1. Open the MashZone NextGen Admin Console.
2. Click **Import/Export** -> **Import/Export** in the **Administration** menu.
3. Select the artifact type you want to import into MashZone NextGen.
4. Enable the **Overwrite if existing** option to overwrite the same already existing artifacts. By default, the import is rejected if the artifacts already exist in MashZone NextGen.
5. Enable the **Import permissions** option to also import the permissions of the selected artifacts.
6. Click **Select a ZIP file** to select the file containing the artifacts you want to import.
7. Click **Test run** to start a test run, to check for possible conflicts with different MashZone NextGen editions.
8. Click **Import** to import the selected artifacts.

The selected artifacts are imported.

When the import is completed successfully, you can use the imported artifacts in MashZone NextGen.

## 5.5 Clustering MashZone NextGen Servers

In production environments, it is common to use clustering solutions to provide better performance for various loads, to provide high availability or to provide both. Because MashZone NextGen is a web application, using an HTTP session based on J2EE standards, you can apply the same cluster architectures and solutions to MashZone NextGen that you use with other web applications.

See Setting Up a New Cluster (page 107) or Adding New Members to an Existing Clusters (page 109) for the tasks you need to complete.

### 5.5.1 Setting Up a New Cluster

The configuration and deployment of a new cluster requires these basic steps:

- Setting Up an External MashZone NextGen Configuration Folder (page 111): this allows you to keep most of the configuration and extensions for MashZone NextGen in a single set of folders that can be shared across the entire cluster. This simplifies both the initial configuration as well as ongoing updates and deployment of new mashables, mashups or apps.

  **Note:** This step is highly recommended, but not required. If you do not use a shared configuration folder, all subsequent updates to configuration or extensions for new artifacts must be manually copied to each member of the cluster.
- This folder should reside in a file system that is shared or mounted across the cluster. You may also need to provide data redundancy or failover capabilities for this shared file system.
- As part of this step, you also typically deploy one MashZone NextGen Server in the cluster and complete most of the basic configuration that will be shared across the cluster.
- Sharing the MashZone NextGen Repository in Clustered Environments (page 110): all nodes in the cluster work with a shared MashZone NextGen Repository which you must create and configure.
- Sharing the MashZone NextGen Repository does not, by itself, provide any data redundancy, load balancing or failover capabilities for the database. These requirements are handled in the data layer by your database server or other replication/synchronization solutions, such as DRBD. For more information, see documentation for your database or replication/synchronization solution.

- **Configuring Caching for the Cluster**: each MashZone NextGen Server has a local cache for mashable and mashup responses as well as local caches for updates to artifacts. If MashZone NextGen Analytics is enabled in your MashZone NextGen license, the MashZone NextGen Analytics In-Memory Stores are also local.
- In clusters you:
- Can leave the response cache as a local cache or you can configure a distributed cache that all MashZone NextGen Servers in the cluster share.
- **Must** configure a distributed cache for artifact updates that all MashZone NextGen Servers in the cluster share.
- **Must** configure a distributed cache for the MashZone NextGen Analytics In-Memory Stores that all MashZone NextGen Servers in the cluster share.
- See Configure BigMemory Max Servers for MashZone NextGen Caching and In-Memory Stores (page 66) for instructions on how to configure BigMemory Max, or other caching solutions, as a distributed cache for .MashZone NextGen
- **Defining the Application Server Cluster**: the application servers that host each MashZone NextGen Server define and handle clustering requirements at the application layer. You can also add a load balancer to the cluster.
- In addition to the basic cluster configuration required by your application server and load balancer, MashZone NextGen has a single requirement for application-layer cluster configuration. You must either:
- Enable session replication in each application server in the cluster.
- Enable session affinity, sometimes also called 'sticky sessions,' in the load balancer.
- Or do both.
- See documentation for your application server and/or load balancer for information on how to do this.
- **Adding Additional MashZone NextGen Servers to the Cluster**: once you have set up the shared resources, you can deploy and add additional members to the cluster. See Adding New Members to an Existing Cluster (page 109) for instructions.
- **Add MetaData and Deploy Artifacts**: for this new environment. For artifacts, you can automate some parts of this process using export and import commands. See Deploying MashZone NextGen Artifacts and Other Metadata (page 93) for instructions.

## 5.5.2 Adding New Members to an Existing Cluster

**To add additional MashZone NextGen Servers to an existing cluster**

1. Install the MashZone NextGen Server. See Installing Software AG Products for instructions.
2. Configure the MashZone NextGen Server to use the shared MashZone NextGen Repository for the cluster. See Share an Existing MashZone NextGen Repository (page 110) for instructions.
3. If the cluster has a shared external configuration folder, add this folder and any subfolders to the classpath for the MashZone NextGen Server's application server to enable access to this shared configuration.

   Depending on your application server, you may update the classpath in the administration console, in configuration files or in the startup script for the application server. See documentation for your application server for more information.
4. If the cluster does not have a shared external configuration folder, copy the configuration and extension files from an existing MashZone NextGen Server in the cluster to the new MashZone NextGen Server.

   See MashZone NextGen File-Based Configuration and Extensions (page 112) for a list of files and folders to copy.
5. Copy the server configuration that cannot be shared from an existing MashZone NextGen Server in the cluster to the new MashZone NextGen Server. See MashZone NextGen File-Based Configuration and Extensionsns (page 112) for details on the files and locations for this step.
6. Update the application server that hosts the new MashZone NextGen Server with the same cluster configuration as other cluster members.

   In addition to the basic cluster configuration required by your application server and load balancer, MashZone NextGen has a single requirement for application-layer cluster configuration. You must either:

   Enable session replication in each application server in the cluster.

   Enable session affinity, sometimes also called 'sticky sessions,' in the load balancer.

   Or do both.

   See documentation for your application server and/or load balancer for information on how to do this.
7. Restart the new MashZone NextGen Server.

## 5.6 Sharing the MashZone NextGen Repository in Clustered Environments

In clustered environments, all MashZone NextGen Servers in the cluster must work with a single, shared MashZone NextGen Repository. You can Create and Share a New MashZone NextGen Repository (page 110) with cluster members, typically when you are creating new environments. Or you can Share an Existing MashZone NextGen Repository (page 110) within a cluster.

### 5.6.1 Create and Share a New MashZone NextGen Repository

**To create a new shared repository**

1. Create a new MashZone NextGen Repository in the appropriate database for your environment. See the table below.
2. Copy the JAR file for the JDBC driver for your database to the **<MashZone NextGen installation>/apache-tomcat/lib** folder on all cluster nodes.
3. If this is a new cluster, update configuration information for the MetaData, User and Snapshot repositories for one MashZone NextGen Server in the cluster. See steps 3 onward in Move the MashZone NextGen repository to a robust database solution (page 24) for instructions.
4. Enable distributed caching for artifacts. See Configure BigMemory Max Servers for MashZone NextGen Caching and In-Memory Stores (page 66) for more information and instructions.
5. If the MashZone NextGen Repository is hosted in Microsoft SQL Server, MySQL or Oracle, change the repository JAR in the MashZone NextGen Server.
6. Restart each MashZone NextGen Server in the cluster.

### 5.6.2 Share an Existing MashZone NextGen Repository

If you are creating a cluster using an existing MashZone NextGen Repository or simply adding members to an existing cluster, you simply update each new MashZone NextGen Server in the cluster to use the existing repository.

**Procedure**

1. If the cluster does not have a shared JDBC driver folder and a shared external configuration folder. Copy the JAR file for the JDBC driver for your database to the MashZoneNG-install/apache-tomcat/lib folder for the new MashZone NextGen Server cluster member.

   See Setting Up an External MashZone NextGen Configuration Folder (page 111) for more information on shared configuration for clusters.
2. Enable distributed caching for artifacts (required) and optionally distributed caching for mashable/mashup responses. See Configure BigMemory Max Servers for MashZone NextGen Caching and In-Memory Stores (page 66) for more information and instructions.
3. If the MashZone NextGen Repository is hosted in Microsoft SQL Server, MySQL or Oracle, change the repository JAR in the MashZone NextGen Server.
4. Restart the new MashZone NextGen Server for this cluster.

## 5.7 Setting Up an External MashZone NextGen Configuration Folder

Most configuration for MashZone NextGen and most of the extensions that you add for your organization's use are stored in the MashZone NextGen Repository. However, some MashZone NextGen configuration and extensions are file based.

By default, MashZone NextGen keeps configuration and extensions in the MashZone NextGen Server web application in these folders:

- <MashZone NextGen installation/apache-tomcat/mashzone/WEB-INF/classes for class, configuration and extension files
- <MashZone NextGen installation/apache-tomcat/mashzone/WEB-INF/lib and
- <MashZone NextGen installation/apache-tomcat/mashzone/WEB-INF/config for JAR files.

You can move most of these configuration and extension files to folders that are external to the MashZone NextGen Server.

**Important:** MashZone NextGen documentation refers to all of these folders as MashZoneNG-config.

Using external configuration folders for MashZone NextGen is a best practice as they simplify deployment and upgrades of the MashZone NextGen Server. They also simplify configuration management for clustered environments. External configuration folders are not required, however.

**To create and use an external configuration folder for MashZone NextGen**

- Create the top-level external folder to use for MashZone NextGen configuration, such as **PrestoConfig**. In clustered environments, share or mount this folder across the entire cluster.
- You can create subfolders under this external folder to organize configuration and extensions.
- For clustered environments, create subfolders under the top-level external configuration folder for:
- The standard **classes** and **lib** folders.
- Built-in and user-defined functions for use in RAQL queries for MashZone NextGen Analytics. See Configure, Compile, Deploy and Test User-Defined Functions for more information.
- If not complete, finish configuration for the MashZone NextGen Server and move the configuration and extension files to the external configuration folder or an appropriate subfolder. See the MashZone NextGen File-Based Configuration and Extensions (page 112) section for the specific configuration steps, files and locations.
- Add the external MashZone NextGen configuration folder, **and any subfolder** that contains extensions or JAR files, to the classpath for the application server(s) hosting the MashZone NextGen Server.
- You may update the classpath in configuration files or in the startup script for the application server.
- For Windows environments, for example, you can edit the tomcat-install/bin/setenv.bat file and update the classpath environmental variable to be something like this:
- **set "CLASSPATH=%CLASSPATH%;C:\PrestoConfig;C:\PrestoConfig\classes;C:\PrestoConfig\lib;C:\PrestoConfig\db\jdbc"**
- On Linux, Mac OS X or UNIX systems, you would update tomcat-install/bin/setenv.sh to something like this:
- **CLASSPATH="$CLASSPATH":/users/PrestoConfig:/users/PrestoConfig/classes:/users/PrestoConfig/lib:users/PrestoConfig/db/jdbc**

## 5.7.1 MashZone NextGen File-Based Configuration and Extensions

Most file-based configuration or extensions involve information that MashZone NextGen needs to connect to the MashZone NextGen Repository or extensions that must be added to

the application server's classpath. In clustered environments, you can share extensions and some of this file-based configuration using an external configuration folder. See MashZone NextGen Configuration Files That Can Be External (page 113) and MashZone NextGen Extensions (page 115) for details on resources that can be shared across a cluster.

Some file-based configuration, however, **must** reside in the web application for each MashZone NextGen Server. In clusters, this configuration must be replicated in each cluster member. See MashZone NextGen Configuration Files That Must Be Internal (page 114) for details.

## 5.7.2 MashZone NextGen Configuration Files That Can Be External

| File | Description and Configuration | Default Location |
|---|---|---|
| **dynamiccache.xml** | Default configuration information for dynamic In-Memory Stores created by MashZone NextGen Analytics. | MashZoneNG-install/apache-tomcat/mashzone/WEB-INF/classes |
| **ehcache.xml** | Configuration information for MashZone NextGen caches. This also contains configuration for MashZone NextGen Analytics In-Memory Stores from version 3.6. | |
| **presto.config** | Miscellaneous MashZone NextGen properties, including the path to the deployed web app home folder. | |
| The BigMemory Max license file | The license file for BigMemory Max, used for MashZone NextGen caches and MashZone NextGen Analytics In-Memory Stores, is a separate license file from the MashZone NextGen license. You can keep the BigMemory Max license in an external folder shared across the cluster.<br>See Configure BigMemory Max Servers for MashZone NextGen Caching and In-Memory Stores (page 66) for required configuration steps to enable a shared license. | |

| File | Description and Configuration | Default Location |
|---|---|---|
| **userRespositoryLdap.properties** | Connection information for your LDAP Directory. See Integrate Your LDAP Directory with MashZone NextGen (page 32) for details. | |

### 5.7.3 MashZone NextGen Configuration Files That Must Be Internal

The file-based configuration that must remain in each MashZone NextGen Server web application resides in the web-apps-home/mashzone/WEB-INF/classes folder.

For upgrades to new MashZone NextGen versions, you can generally copy these configuration files from your existing MashZone NextGen version to the new version. Review the MashZone NextGen Release Notes for changes or new features that may require updates to configuration.

For clustered environments, you **must** copy these configuration files to each cluster member. In most cases, you change configuration once, when you first deploy a MashZone NextGen Server in the cluster. Any subsequent changes to this configuration for one cluster member, however, must be copied to all other cluster members manually, using a scheduled job or using another replication scheme.

| File | Description and Configuration |
|---|---|
| **applicationContext-commonServices.xml** | You edit configuration in this file if you choose to use distributed response caching for MashZone NextGen. See Configure BigMemory Max Servers for MashZone NextGen Caching and In-Memory Stores (page 66) for more information.<br>You may need to update this configuration, as needed, to add additional distributed cache nodes to tune performance. |
| **applicationContext-security.xml** | You edit this file initially to enable either SSO authentication or X509 certificate authentication for MashZone NextGen. See Authentication with single sign-on solutions (page 8) for more information. |
| **log4j.properties** | This file is updated automatically when you change logging configuration in the Admin Console. See Configure Logging for the MashZone NextGen Server (page 61) for details.<br>When you change logging for MashZone NextGen Servers in a cluster, |

| | |
|---|---|
| | only the specific MashZone NextGen Server that the Admin Console is connected to is affected. To change logging for the entire cluster, you must update this file and copy it to each cluster member. |
| **userRepositoryApplicationContext.xml** | You edit these files when you configure MashZone NextGen to use your LDAP Directory as the user repository. See Integrate Your LDAP Directory with MashZone NextGen (page 32) for details. |
| **userRepositoryApplicationContext-ldap.xml** | |

### 5.7.4 MashZone NextGen Extensions

Some extensions, such as macros, are registered and reside in the MashZone NextGen Repository. Any of the following file-based extensions can reside in an external folder:

| File | Default Location |
|---|---|
| Custom filter classes for single sign-on authentication. See Implementing a Custom SSO Filter (page 11) for details. | <MashZone NextGen installation>/apache-tomcat/mashzone/WEB-INF/classes |
| Classes and third-party libraries for a user-defined function library to use with RAQL. See Create and Add User-Defined Functions for RAQL Queries for more information. | or <MashZone NextGen installation>/apache-tomcat/mashzone/WEB-INF/lib (for JARs) |

## 5.8 MashZone NextGen dashboards in a clustered scenario

You can use MashZone NextGen dashboards in a clustered scenario.

The following chapters describe how to configure MashZone NextGen to use dashboards and data feeds in a multiple master-client scenario.

### 5.8.1 Preliminary

Before you can configure MashZone NextGen using in a clustered scenario you have to perform the following steps.

**Procedure**

1. Install at least two regular MashZone NextGen instances on two different machines.

   Software AG Installer enables you to install MashZone NextGen. Detailed information on how to use Software AG Installer is available in the documentation **Using the Software AG Installer**.
2. Connect all instances to the same central database according to section Move the MashZone NextGen repository to a robust database solution (page 24).

The preliminary for configuring MashZone NextGen are completed.

### 5.8.2 Configuration

The following chapters describe the relevant configurations of MashZone NextGen dashboards in a clustered scenario.

### 5.8.3 MashZone NextGen nodes

Edit the **mashzone.feedprocessing.timezone** property in the **mashzone.properties** file and specify a common timezone in all nodes of the cluster.

<MashZone NextGen installation>/apache-tomcat/webapps/mashzone/WEB-INF/mashzone.properties

For more details, see the chapter **Configure feed processing time zone** (page 46).

**Example**

mashzone.feedprocessing.timezone=UTC

Use UTC, or replace it by the desired time zone ID. You can only use time zones that are compatible with the Java **TimeZone.getTimeZone** method.

#### 5.8.3.1 Customizing dashboards

MashZone NextGen dashboards can be customized by adding custom style templates for the dashboard application and the dashboard content. Additionally custom widgets can be created via the pluggable widget framework. If these options shall be applied in a clustered

scenario, you must synchronize the relevant folders and restart MashZone NextGen on all nodes of the cluster.

### 5.8.4 Custom styles

By default, custom style templates available are stored in the following folders.

- <MashZone NextGen installation>/apache-tomcat/webapps/mashzone/hub/dashboard/assets/custom-look-and-feel/application
- <MashZone NextGen installation>/apache-tomcat/webapps/mashzone/hub/dashboard/assets/custom-look-and-feel/dashboard

To apply the custom templates on all cluster nodes, make sure that these folders are synchronized on all machines. Since the less files need to be compiled before the styles can be used, MashZone NextGen has to be restarted on all cluster nodes.

### 5.8.5 Custom widgets

By default, custom widgets available are stored in the following folders.

<MashZone NextGen installation>/apache-tomcat/webapps/mashzone/hub/dashboard/widgets/customWidgets

To make the custom widgets available on all cluster nodes, make sure that the folders is synchronized on all machines. In this case, restarting MashZone NextGen on all cluster nodes is required as well.

### 5.8.6 Using JDBC drivers

JDBC driver binaries have to be available on every cluster node to allow class loading in the JVM. Since MashZone NextGen version 9.10 the binaries are stored in the DB and restored in <MashZone NextGen installation>\apache-tomcat\webapps\mashzone\WEB-INF\config\db\jdbc on all cluster nodes if not available. Automatic class loading on demand works fine, so that no further steps have to be taken to make JDBC resources available in a clustered scenario.

### 5.8.7 Local file resources

Local file resources are not recommended and not supported in a clustered scenario due to synchronization issues.

In a Windows landscape it might be possible to use such file resources by mapping the same network drive to the same network share. There may also be other file sharing mechanisms working in other OS landscapes, but URL based access is preferable.

## 5.9 Customize application and dashboard styles

You can customize the styles of the MashZone NextGen application, dashboards, and widgets. For example, you can replace the brand logo and the text displayed on the MashZone NextGen welcome page. You can edit the look and feel of the dashboards and widgets, for example, colors schemes, fonts, or background colors. You can use the available styles in MashZone NextGen to create your own style templates.

MashZone NextGen provides two modes to customize the application and dashboard styles. For details, see the chapter Switch the style editing mode (page 124).

### 5.9.1 Customize dashboard and widget styles

You can create and add your own styles to customize the look and feel of your dashboards and widgets.

**Tip**

You can use the **default** style as template for your own styles. MashZone NextGen styles are stored in the database and can be downloaded and saved as LESS style files. (page 121)

In file-based mode (page 124), you can edit the LESS files stored in the following folder.

<MashZone NextGen installation\apache-tomcat\webapps\mashzone\hub\dashboard\assets\custom-look-and-feel\dashboard

**Prerequisite**

You have administration privileges.

**Procedure**

1. Open a LESS style file in an appropriate text editor.
2. Specify your style settings.
3. Save your settings.

4. In database mode, upload the modified style file (page 121). You can replace an available style or add a new style. The styles are uploaded if they can be compiled by the MashZone NextGen server. In database mode, the styles are automatically compiled by default.
5. In file-base mode, perform the following steps.
    a. If you want to modify an available style, replace the corresponding LESS file with the modified file.
    b. If you want to add a new style, create a folder as described in the chapter How are the style files structured? (page 127), and store the modified LESS file in the new folder with a new file name.
    c. Compile the modified style. (page 122)

The new style is now available in the MashZone NextGen.

In the dashboard editor, you can select the uploaded styles to change the look and feel of your dashboards and widgets. For details, see the chapters Change the dashboard style and Change the widget style.

## 5.9.2 Customize the application style

You can customize the look and feel of the entire application. Your changes of the application styling affect the MashZone NextGen welcome page (page 120), the dashboard editor in view and edit mode, and the feed editor. The styling of the Admin Console cannot be changed.

The styling information of the entire application are specified in the **application.less** file.

### Prerequisites

You have MashZone NextGen administrator privileges.

### Procedure

1. In database mode, perform the following steps.
    a. Download the **application.less** file. (page 121)
    b. Edit the styles in an appropriate text editor.
    c. Save your settings.
    d. Upload the modified file. (page 121)
2. In file-based mode, perform the following steps.
    a. Edit the **application.less** style file located in the following folder.

        <MashZone NextGen installation>\apache-tomcat\webapps\mashzone\hub\dashboard\assets\custom-look-and-feel\application
    b. Compile the modified **application.less** file. (page 122)

Your changes are applied.

### 5.9.3 Customize the MashZone NextGen welcome page

You can replace the logo and the welcome text displayed on the MashZone NextGen welcome page. In addition, you can change the style of the welcome page header.

You must replace the relevant logo and welcome text on each hosting MashZone NextGen server.

Your changes in the welcome page header are applied to all application headers in MashZone NextGen, also to the dashboard and data feed editor.

**Prerequisite**

You have administration privileges.

**Procedure**

1. To replace the logo on the MashZone NextGen welcome page, you must replace the **landing_page_icon.png** graphic file in the following folder by your own graphic file.

   <MashZone NextGen installation>\apache-tomcat\webapps\mashzone\hub\assets\images\

   Alternatively, you can specify a path to an image file using the **@brand-logo** parameter in the **application.less** file. The path must be absolute and begin with "/".
2. To replace the welcome text on the MashZone NextGen welcome page, open the **welcome-text.json** file with a text editor. Enter a new text and save your changes. The file is located in the following folder.

   <MashZone NextGen installation>\apache-tomcat\webapps\mashzone\WEB-INF\config\
3. To change the appearance of the application header, you can edit the style templates supplied with MashZone NextGen. For details, see the chapter **Edit style templates**.
4. You can upload the customized **application.less** file in the Admin Console if you use the database mode. For details, see the chapter **Upload styles** (page 121).
5. If you use the file-based mode, perform the following steps.
   a. Copy the customized **application.less** file in the following folder.

      <MashZone NextGen installation>\apache-tomcat\webapps\mashzone\hub\dashboard\assets\custom-look-and-feel\application
   b. Compile the modified application styles (page 122).

Your changes are applied.

#### MIGRATION OF THE APPLICATION.LESS FILE

If you have migrated the **application.less** file from MashZone NextGen version 9.12 or older, and you have customized the application header, you must adapt the **application.less** file of your current version.

You must change the path reference of the logo used in the application header. The path reference in the **@brand-logo** key must be absolute and starts with "/", for example, '/hub/dashboard/assets/images/my-logo.png'.

Add the following key.

```
// Font size used for application menu items in the dropdown menu of the masthead
@navigation-list-font-size-menu-dropdown: 14px;
```

## 5.9.4 Download styles

You can download available styles to use as style templates.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Styles & Themes** to expand this section of the **Administration** menu.
3. Click **Styles and Themes** to open the corresponding page.
4. To download the style file of the application, open the **Application** page and click **Download**.

   The **application.less** style file is downloaded.
5. To download a dashboard or widget style file, open the **Dashboard** page and click the **Download** icon of the desired style file.

The selected style file is downloaded and stored in the download folder specified in your web browser.

## 5.9.5 Upload styles

You can upload custom styles for your dashboards, widgets, and the entire application. The styles are stored in the MashZone NextGen database.

If you want to change or add a dashboard style, just upload the corresponding LESS style file, for example, myDashboardStyle.less. The LESS file can also be stored in a ZIP file. The ZIP file can have any name, for example, myStyles.zip.

If you want to change or add a dashboard style and widget styles, or just widget styles, you must upload a ZIP file that contains the corresponding LESS style files. The style files must be stored in the required folders, and the ZIP file must contain a corresponding file structure as described in chapter **How are the style files structured?** (page 127).

**Prerequisite**

You have administration privileges.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Styles & Themes** to expand this section of the **Administration** menu.
3. Click **Styles and Themes** to open the corresponding page.
4. To upload styles for the entire application, open the **Application** page.
    a. Click **Upload**.
    b. Select the **application.less** file to be uploaded.
    c. Click **Open**.
5. To upload styles for a dashboard, open the **Dashboard** page.
    a. Click **Upload**.
    b. Select the **LESS** style file or the **ZIP** file that contains the style file to be uploaded.
    c. Click **Open**.
6. To upload styles for a dashboard

The selected styles are uploaded to the MashZone NextGen database.

In the dashboard editor, you can select the uploaded styles to change the look and feel of your dashboards and widgets. For details, see the chapters Change the dashboard style and Change the widget style.

## 5.9.6 Compile styles

If you use the file-based mode (page 124) to customize the styles of your dashboards and widgets, or the entire application, you must manually start the compiling of the modified style files to apply your changes in MashZone NextGen.

If you have modified widget styles, you must compile the corresponding dashboard style to apply your settings.

In database mode (page 124), the styles are automatically compiled when you upload the modified style file (page 121).

Procedure

1. Open the MashZone NextGen Admin Console.
2. Click **Styles & Themes** to expand this section of the **Administration** menu.
3. Click **Styles and Themes** to open the corresponding page.
4. To compile application styles, open the **Application** page and click **Compile**.
5. To compile dashboard and widget styles, open the **Dashboard** page and click **Compile** for the desired styles and themes.

   In file-based mode, the page lists all dashboard styles stored in the file system.

   <MashZone NextGen installation\apache-tomcat\webapps\mashzone\hub\dashboard\assets\custom-look-and-feel**dashboard**

   You can only compile one style at a time. While compiling a style, all **Compile** buttons are disabled.

The selected styles are compiled.

#### CUSTOM WIDGETS

If you modify styles of a custom widget (page 126) while the server is started, you must manually start the compiling of the corresponding styles. Compile the dashboard styles that you want the custom styles to contain, such as the default dashboard style. In database mode, click the Compile icon of the desired dashboard style.

## 5.9.7 Delete styles

You can delete dashboard and widget styles from the database.

**Warning**

Deleted styles and themes cannot be restored.

The application and the default styles cannot be deleted.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Styles & Themes** to expand this section of the **Administration** menu.
3. Click **Styles and Themes** to open the corresponding page.
4. Open the **Dashboard** page and click the **Delete** icon of the desired styles.

The selected styles are deleted from the database.

## 5.9.8 Switch the style editing mode

MashZone NextGen provides the database mode and the file-based mode to customize the application and dashboard styles. You can switch between the two editing modes.

### DATABASE MODE

The database mode is the default mode to customize the look and feel of MashZone NextGen. All styles are stored in the MashZone NextGen database. A style can be downloaded from the database and edited to change it or create a new style. The changed style can then be uploaded back to the database to replace the old style or to add a new one.

### FILE-BASED MODE

The file-based mode is used in a test or development environment to see immediate results of your customizing in MashZone NextGen. You can edit an existing style file, or add a new style file to your local installation. Then you must only compile your changed styles and the result of your changes will take effect immediately.

**Warning**

Switching the mode for a running system may cause unwanted effects. Instead of switching the mode for a running system, we recommend installing a test system to create and test style changes. The production system should always run in database mode.

**Procedure**

1. Open the **presto.config** file in an appropriate text editor. The file is located in the following directory.

   <MashZone NextGen installation\apache-tomcat\webapps\mashzone\WEB-INF\classes\
2. Specify the **enable.stylingconfiguration.fileBasedMode** parameter.

   **false**: enables the database mode. This setting is default.

   **true**: enables the file-based mode.
3. Save your changes.

Your settings are applied.

## 5.9.9 Adjust the font size to different devices

If you display your dashboards on devices with different screen sizes, for example, on mobiles, or tablets, you can adapt the font size of the displayed widgets.

The font size of most widget elements, such as the widget title, is automatically adjusted to the device size. But you can adjust the font size of all widget elements, such as the KPI values of a speedometer (see the example below).

By default, MashZone NextGen uses a predefined font size with the default size factor **1**. To adjust the font size to different devices, MashZone NextGen multiplies the default font size by the following predefined device factors:

| Device | Device factor<br>(Factors for media query) | Font size, for example |
|---|---|---|
| Desktop (default) | @default-factor: 1;<br>min-width: 950px<br>max-width: 1128px | 12px |
| Mobile | @small-device-factor: 0.8;<br>max-width: 529px | 10px |
| Tablet | @medium-device-factor: 0.9;<br>min-width: 530px)<br>max-width: 949px | 11px |
| Large devices | @large-device-factor: 1.2;<br>min-width: 1129px | 15px |

The font sizes are specified in the **default.less** template file. The file is located in the following directory.

<MashZone NextGen installation>\apache-tomcat\webapps\mashzone\hub\dashboard\assets\custom-look-and-feel\dashboard

### Example

You want to specify the font sizes for the KPI values of a speedometer. The font sizes for mobiles and tables are based on the predefined font size **font-big**. **Font-big** is set as default size for the widget element here.

```
@speedometer-kpi-value-font-size:           @font-big;
@speedometer-kpi-value-font-size-mobile:
      ceil(@speedometer-kpi-value-font-size * @small-device-factor);
@speedometer-kpi-value-font-size-tab:
      ceil(@speedometer-kpi-value-font-size * @medium-device-factor);
@speedometer-kpi-value-font-array:
      @speedometer-kpi-value-font-size-mobile
@speedometer-kpi-value-font-size-tab;
```

### Procedure

1. Open the **default.less** style file in your text editor.

2. Add your font size specifications for a widget element, as shown in the example above.
   a. Set the default font size for your widget element.

      For example:

      **@speedometer-kpi-value-font-size: @font-big;**

   b. Set the font size for different devices.

      For example:

      **@speedometer-kpi-value-font-size-mobile:**
      **ceil(@speedometer-kpi-value-font-size * @small-device-factor);**

      **@speedometer-kpi-value-font-size-tab:**
      **ceil(@speedometer-kpi-value-font-size * @medium-device-factor);**

   c. List the font sizes in a font array.

      For example:

      **@speedometer-kpi-value-font-array:**
      **@speedometer-kpi-value-font-size-mobile**
      **@speedometer-kpi-value-font-size-tab;**
3. Save your settings.
4. In database mode, upload the **default.less** (page 121).
5. In **file-based** mode replace the **default.less** file in the following directory and compile the modified style (page 122).

   <MashZone NextGen installation>\apache-tomcat\webapps\mashzone\hub\dashboard\assets\custom-look-and-feel\dashboard

You settings are applied.

## 5.9.10 Custom widgets

You can modify the styles of custom widgets (page 118), such as the Function Flow diagram.

Custom widgets are installed in the following folder:

<MashZone NextGen installation\apache-tomcat\webapps\mashzone\hub\dashboard\widgets\customWidgets

The style files of the custom widgets are automatically installed in corresponding subfolders of the **default** dashboard style.

**Example**

<MashZone NextGen installation\apache-tomcat\webapps\mashzone\hub\dashboard\assets\custom-look-and-feel\dashboards\default\functionFlowWidget

The MashZone NextGen server automatically compiles the styles of newly installed custom widgets when the server starts.

Note that if you modify styles of a custom widget (page 126) while the server is started, you must manually start the compiling of the corresponding styles. In database mode (page 124), click the **Compile** icon of the **default** dashboard style. On how to start the compiling of a dashboard style, see the chapter **Compile styles** (page 122).

## 5.9.11 Style file structure

The different styles are specified in various **LESS** style files, such as application.less or default.less.

### APPLICATION STYLING

The general styling settings of the entire application are specified in the **application.less** file. Changes in the **application.less** file affect the MashZone NextGen welcome page, the dashboard editor in view and edit mode, and the feed editor. The styling of the Admin Console cannot be changed.

The **application.less** file is stored in the following folder.

<MashZone NextGen installation>\apache-tomcat\webapps\mashzone\hub\dashboard\assets\custom-look-and-feel\application

### DASHBOARD AND WIDGET STYLING

The dashboard and widget styles are specified in various style files with specific style names, such as White.less or Default-big.less. The default dashboard style file is named **default.less**.

The dashboard style files are store in the **dashboard** style folder:

<MashZone NextGen installation\apache-tomcat\webapps\mashzone\hub\dashboard\assets\custom-look-and-feel**dashboard**

Example

The LESS files of the **default** and **dark** dashboard styles:

...\assets\custom-look-and-feel\dashboard****

- default.less
- dark.less

By default, a dashboard style provides a set of widget styles, that are stored in the corresponding widget style subfolder. The following folder names can be used to create the relevant widget subfolders: **actionbutton**, **barchart**, **bubblechart**, **colorpalette**, **columnchart**, **container**, **datefilter**, **grid**, **horizontalgauge**, **label**, **layoutgroup**, **layoutrow**, **linechart**, **piechart**, **speedometer**, **tab**, and **verticalgauge**.

Note that the list may vary depending on the MashZone NextGen version and the installed custom widgets.

**Example**

...\dashboard**\default**

**\actionbutton**

**\barchart**

**\bubblechart**

**\colorpalette**

...

For each widget style, a corresponding **LESS** style file is created and stored in the corresponding widget subfolder. The various styles must be assigned to the individual widget.

**Example**

The **Number_left_aligned** is assigned to the **Bar chart** and the **Action button** widget. The **Red** style is only assigned to the **Bar chart** widget.

...\dashboard\default\

- actionbutton\

  **Number_left_aligned.less**
- barchart\

  **Number_left_aligned.less**

  **Red.less**

A widget style assigned to a dashboard applies to all widgets of the same type in this dashboard, for example, all bar charts in the dashboard.

## FILE UPLOAD

You can upload your styles in individual LESS files or contained in a ZIP file. (page 121)

The structure of the ZIP file must correspond to the file and folder structure described above. The ZIP file can have any name.

**Example**

myStyling.zip

- dashboardStyle.less
- dashboardStyle
  - actionbutton
    - Default-big.less
    - Primary-big.less
    - Primary.less
  - barchart
    - White.less
  - bubblechart
    - White.less
  - ...
  - grid

## 5.10 Empty MashZone NextGen caches

You can manually delete the data of the internal caches that MashZone NextGen uses for caching the results of processing data sources and data feeds. By default, the caches are automatically emptied when the configured data refresh rate has expired.

MashZone NextGen provides a specific service to empty the cache on your local MashZone NextGen installation and the caches on all nodes in a multi-node environment remotely. You cannot empty the caches of individual nodes or a set of nodes in a multi-node environment.

**Prerequisites**

You have administration privileges.

**Procedure**

1. Start MashZone NextGen.
2. Log in to MashZone NextGen with administration privileges.
3. Enter the following URL in the address bar of your Web browser and press **Enter**.

   http://<hostname>:<port>/mashzone/mzservices/admin/clearresultcaches

All caches have been emptied. A corresponding note informs you when the caches have been successfully emptied.

# 6 Event Service Configuration and Administration

## 6.1 Manage Apama Instances

By creating an Apama instance (Apama correlator) you can specify the connection to an Apama system.

You can create, edit and delete instances.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Event Service** to expand this section of the **Administration** menu.
3. Click **Apama instances**.
4. Select further steps:

   Create Apama Instances (page 130)

   Edit Apama Instances (page 131)

   Delete Apama Instances (page 131)

### 6.1.1 Create Apama Instances

Creating an Apama instance (Apama correlator) you can specify the connection to an running Apama system.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Event Service** to expand this section of the **Administration** menu.
3. Click **Apama instances**.
4. Click **Create**.
5. Set the properties of the Apama instance. See table below.
6. Click **Save**.

The Apama instance is created and listed by alias name.

**Apama instance properties**

| Property | Required | Description |
|---|---|---|
| Alias | yes | Enter a unique name for this Apama instance. |
| Host | yes | Host name to the running Apama system ( local or remote) |

| | | |
|---|---|---|
| Port | yes | Port number of the running Apama system ( local or remote) |

## 6.1.2 Edit Apama Instances

You can edit an already existing Apama instances.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Event Service** to expand this section of the **Administration** menu.
3. Click **Apama instances**.
4. Click the **Edit** icon to configure an Apama instance.
5. Set the properties of the Apama instance. See table below.
6. Click **Save**.

Your changes are applied.

**Apama instance properties**

| Property | Required | Description |
|---|---|---|
| Alias | yes | Enter a unique name for this Apama instance. |
| Host | yes | Host name to the running Apama system ( local or remote) |
| Port | yes | Port number of the running Apama system ( local or remote) |

## 6.1.3 Delete Apama Instances

You can delete Apama instances.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Event Service** to expand this section of the **Administration** menu.
3. Click **Apama instances**.
4. Click the **Delete** icon to delete a specific Apama instance.

The selected Apama instance is deleted from the list.

## 6.2 Manage Apama Event Targets

An Apama event target specifies an Apama system that can receive events sent by MashZone NextGen.

You can create, edit and delete Apama event targets.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Event Service** to expand this section of the **Administration** menu.
3. Click **Event Service**.
4. Open the **Apama** tab.
5. Select further steps:

   Create Apama Event Targets (page 132)

   Edit Apama Event Targets (page 133)

   Delete Apama Event Targets (page 134)

   Share Apama Event Target (page 134)

### 6.2.1 Create Apama Event Targets

Creating an **Apama** event target you can specify an **Apama** system as target, receiving events from MashZone NextGen.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Event Service** to expand this section of the **Administration** menu.
3. Click **Event Service**.
4. Open the **Apama** tab.
5. Click **Create Apama Event Targe**t.
6. Set the properties for this event target. See table below.
7. Click **Save**.

The Apama event target is created and listed by alias name.

| Property | Required | Description |
|---|---|---|
| Alias | yes | Enter a unique name for this event target. |

| | | |
|---|---|---|
| **Apama** instance | yes | Alias with the pre-configured connection specification of a running **Apama** system (local or remote). See Manage Apama Instances (page 130) for details. |
| Event type | yes | Click **Refresh** to update the list of Apama event types. Only event types are available that are present in the Apama instance selected. Select the type of the event this event target should subscribe to.<br>Event types including not supported data types by MashZone NextGen are also available; these event types can be used for sending events, but the offending fields are not usable for data assignment to widgets. |

## 6.2.2 Edit Apama Event Targets

You can edit an already existing Apama event target.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Event Service** to expand this section of the **Administration** menu.
3. Click **Event Service**.
4. Open the **Apama** tab.
5. Click the **Edit** icon to configure an Apama event target.
6. Set the properties for this event target. See table below.
7. Click **Save**.

Your changes are applied.

**Apama event target properties**

| Property | Required | Description |
|---|---|---|
| Alias | yes | Enter a unique name for this event target. |
| **Apama** instance | yes | Alias with the pre-configured connection specification of a running **Apama** system (local or remote). See Manage Apama Instances (page 130) for details. |
| Event type | yes | Click Refresh to update the list of Apama event types. Select |

| | | |
|---|---|---|
| | | the type of the event this event target should subscribe to.<br>Event types including not supported data types by MashZone NextGen are also available; these event types can be used for sending events, but the offending fields are not usable for data assignment to widgets. |

### 6.2.3 Delete Apama Event Targets

You can delete Apama event targets.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Event Service** to expand this section of the **Administration** menu.
3. Click **Event Service**.
4. Open the **Apama** tab.
5. Click the **Delete** icon to delete a specific Apama event target.

The selected Apama event target is deleted from the list.

### 6.2.4 Share Apama Event Target

You can share Apama event targets with particular users and user groups so that these have access to Apama event targets.

**Prerequisite**

You have administration privileges.

Regardless of the share, users with administration privilege can access all Apama event targets.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **Event Service** to expand this section of the **Administration** menu.
3. Click **Event Service**.
4. Open the **Apama** tab.
5. Click the **Edit permissions** icon of the Apama event target you want to share.

6. Enter a term in the search field and click **Search**. Clicking on **Search** without any input values fetches all users and groups.
7. Drag an user or user group from the **Search result** field and drop it into the **Principals with permissions** field.

   By default, the owner of the **Apama Event Services** is already present in the **Principals with permissions** list . This owner is non editable and cannot be removed from the list.
8. Activate or deactivate the Display or Usage privileges of a user or user group.

   A user or user group with **View** privilege can see the relevant source data in the data feed or dashboard. A user or user group with the **Edit** privilege has access to the relevant alias in the data source operator.
9. Click **OK**.

Your changes are applied.

# 7 Process Performance Manager Integration

ARIS Process Performance Manager (PPM) lets you discover and analyze processes that are not formally managed by a business process management solution (**BPMS**), such as webMethods BPMS. Using data sources throughout your enterprise, such as transactional data from your business systems, event streams from webMethods BPMS or database records from trading partners, PPM can model a process and assess its performance across various dimensions, such as region, product line, volume, or time. You can also use PPM analytic tools to mine other data in your enterprise for meaningful patterns, trends, or correlations.

Information from PPM can be used as a source of data for MashZone NextGen dashboards and data feeds.

MashZone NextGen is compatible with PPM version 10.0 or above.

## 7.1 Manage PPM Connections

You can manage your PPM Connections in the **Admin console**.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **PPM connections** to expand this section of the **Administration** menu.
3. Click **PPM connections**.
4. Follow the procedure of the remaining steps:

- Create PPM Connections (page 136)
- Edit PPM Connections (page 138)
- Delete PPM Connections (page 139)
- Share PPM connections (page 139)

## 7.2 Create PPM Connections

You define connections for one or more PPM clients to allow users to use PPM as a data source for MashZone feeds or to allow users to add charts from PPM to workspace apps in MashZone NextGen.

MashZone NextGen is compatible with PPM 10.0 or above.

For MashZone NextGen to connect and retrieve PPM data or charts, the following PPM applications must be started:

- PPM
- PPM client

For details on your PPM installation, contact the system administrator in charge.You can enter PPM connection information manually or you can have MashZone NextGen determine them using the URL of a PPM favorite (favorites path). For information on copying the URL of a PPM favorite, see PPM online help topics.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **PPM connections** to expand this section of the **Administration** menu.
3. Click **PPM connections**.
4. Click **Create**.
5. Enter a name for the PPM connections in the Alias field, for example, the client name. The connection data is saved under this alias. Users may choose PPM connections by their alias.
6. To retrieve the connection data from the URL of a favorite from PPM:
   a. Click **Retrieve data**.
   b. Enter the URL of the PPM favorite that you copied earlier in the URL field.
   c. Click **Resolve URL** to retrieve the required parameters from the URL. MashZone NextGen uses the favorite URL to complete the remaining fields for this connection.
7. To enter connection information manually:
   a. Select the protocol (HTTP or HTTPS) to use for the web application server that hosts the PPM query interface.

      For safety reason, we recommend using the HTTPS protocol.
   b. In the **Host** field, enter the fully qualified domain name of the PPM load balancer.
   c. In the **Port** field, enter the port number of the PPM load balancer.
   d. Specify the PPM client name of your PPM connection in the **Client** field.
8. Click **Check availability** to verify that the data is correct and that the PPM client is available.
9. Click **Save**.

The PPM connection is created and listed by alias name. This also lists the PPM version and availability of the PPM client.

## 7.3 Edit PPM Connections

You can edit already existing PPM connections.

Changes in PPM connection properties can immediately affect data feed calculations so that they may not execute properly.

For MashZone NextGen to connect and retrieve PPM data or charts, the following PPM applications must be started:

- PPM
- PPM client

For details on your PPM installation, contact the system administrator in charge. You can enter PPM connection information manually or you can have MashZone NextGen determine them using the URL of a PPM favorite (favorites path). For information on copying the URL of a PPM favorite, see PPM online help topics.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **PPM connections** to expand this section of the **Administration** menu.
3. Click **PPM connections**. A list of all available PPM connections is displayed.
4. Click the **Edit** icon to configure a PPM connection.
5. The **Alias** field of an already configured PPM connection is not editable. The connection data is saved under this alias.
6. To retrieve the connection data from the URL of a favorite from PPM:
   a. Click **Retrieve data**.
   b. Enter the URL of the PPM favorite that you copied earlier in the **URL** field.
   c. Click **Resolve URL** to retrieve the required parameters from the URL. MashZone NextGen uses the favorite URL to complete the remaining fields for this connection.
7. Enter the connection information manually.
   a. Select the protocol (HTTP or HTTPS) to use for the web application server that hosts the PPM query interface.

      For safety reason, we recommend using the HTTPS protocol.
   b. In the **Host** field, enter the fully qualified domain name of the PPM load balancer.
   c. In the **Port** field, enter the port number of the PPM load balancer.
   d. Specify the PPM client name of your PPM connection in the **Client** field.
8. Click **Check availability** to verify that the data is correct and that the PPM client is available.
9. Click **Save**.

Your changes are applied.

## 7.4 Delete PPM Connections

You can delete existing PPM connections.

**Warning**

Deleting PPM connections may cause data feeds to fail.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **PPM connections** to expand this section of the **Administration** menu.
3. Click **PPM connections**. A list of all available PPM connections is displayed.
4. Click the **Delete** icon to delete a PPM connection.
5. Confirm the deletion.

The selected PPM connections are deleted from the list.

## 7.5 Share PPM connections

You can share PPM connections with particular users and user groups so that these have access to PPM server.

You have administration privileges.

Regardless of the share, users with administration privilege can access all PPM connections.

**Procedure**

1. Open the MashZone NextGen Admin Console.
2. Click **PPM Connections**.
3. Click the **Edit PPM alias permissions** icon of the PPM connection you want to share.
4. Enter a term in the search field and click **Search**. Clicking on **Search** without any input values fetches all users and groups.
5. Drag an user or user group from the **Search result** field and drop it into the **Principals with permissions** field.

   By default, the owner of the PPM connection is already present in the **Principals with permissions** list . This owner is non editable and cannot be removed from the list.
6. Activate or deactivate the **Display** or **Usage** privileges of a user or user group.

A user or user group with **Display** privilege can see the relevant source data in the data feed or dashboard. A user or user group with the **Usage** privilege has access to the relevant alias in the data source operator.

7. Click **OK**.

Your changes are applied.

# 8 webMethods Business Console Integration

MashZone NextGen can be easily embedded in webMethods Business Console using a native Business Console gadget.

The gadget is called MashZone NextGen and can be found in the Business Console **Common** section.

Detailed information on how to use webMethods Business Console can be found in the documentation **Working with webMethods Business Console**.

In order to access a MashZone NextGen dashboard, the **Dashboard URL** must be provided in the gadget settings. The URL must contain the MashZone NextGen dashboard GUID as an URL parameter.

## 8.1 Example

http://sbrvpresto4.eur.ad.sag:8080/mashzone/hub/dashboard/dashboard.jsp?guid=e35c1619-0b06-42ac-b343-b16e7d5dcc12

In the section **UI Settings** the gadget height, title and border can be specified.

The section **Data Mapping** specifies the parameters required for the communication between MashZone NextGen dashboards and Business Console gadgets.

- **Mapping Id**: Identifier used in Business Console for gadget to gadget communication. The **Mapping Id** is needed to identify and map the data send from one gadget to the data structure of another gadget. In case of two MashZone NextGen gadgets, exchanging data, the mapping can be used to take a selection value from one embedded MashZone NextGen widget and use it as selection value in the other MashZone NextGen widget.
- **Widget Id**: Specifies the external identifier of the MashZone NextGen widget to communicate with.
- **Widget Parameter**: Specifies a measure or dimension name used in the MashZone NextGen widget.
- **Default Value**: Optionally, it is possible to define a default value, that is used for example as a default selection for the MashZone NextGen widget after loading the gadget in Business Console.

All data required can be found in MashZone NextGen, see **Use dynamic URL selection** for details.

## 8.2 Authentication

You can integrate MashZone NextGen under My webMethods in an SSO scenario by SAML (Security Assertion Markup Language).

MashZone NextGen can accept SAML tokens for authentication in a SSO environment.

See Authentication with single sign-on solutions (page 8) for details.

A BASE64 encoded SAML token is expected. Since it is send via URL it needs to be URL encoded before.

## 8.3 Example URL

http://sbrvpresto4.eur.ad.sag:8080/mashzone/hub/dashboard/dashboard.jsp?appheader=false&guid=64545b4f-d150-4241-a858-2304eea23684 &SAMLToken=<URL encodedBASE64 encoded token>

## 8.4 Configuration

To enable access to MashZone NextGen you need to list the URL(s) of the webMethods Business Console server(s) in the Content Security Policy of MashZone NextGen.

The content security settings are done in the server configuration file **applicationContext-security-filters.xml** by adding filters for X-Frame-Options and Content Security Policies. The file is located in <MashZone NextGen installation>\apache-tomcat\webapps\mashzone\WEB-INF\classes.

**applicationContext-security-filters.xml (abstract)**

```
<beans:beans
xmlns="http://www.springframework.org/schema/security"...>
...
<http pattern="/hub/(login|reset_password)\.html.*" security="none"
request-matcher="regex"/>
<http pattern="/help/.*" security="none" request-matcher="regex"/>
<http pattern="/**/*.jsp" use-expressions="false"
authentication-manager-ref="authenticationManager"
entry-point-ref="mzngAuthenticationEntryPoint">
<anonymous enabled="false"/>
<headers>
<!--frame-options policy="SAMEORIGIN"/-->
<frame-options policy="ALLOW-FROM" strategy="static"
value="http://otherServerHost:otherServerPort" />
<!--content-security-policy policy-directives="frame-ancestors 'self'"/-->
<content-security-policy policy-directives="frame-ancestors 'self'
http://otherServerHost:otherServerPort"/>
</headers>
```

```
<csrf token-repository-ref="csrfTokenRepository"
request-matcher-ref="skipHttpAuthCsrfMatcher"/>
<custom-filter ref="samlTokenProcessingFilter" after="PRE_AUTH_FILTER"/>
<custom-filter ref="jwTokenProcessingFilter" before="CAS_FILTER"/>
<custom-filter ref="credentialContainerFilter"
before="EXCEPTION_TRANSLATION_FILTER"/>
</http>
<http pattern="/**/*.html" use-expressions="false"
authentication-manager-ref="authenticationManager"
entry-point-ref="mzngAuthenticationEntryPoint">
<intercept-url pattern="/**/*.html"
access="IS_AUTHENTICATED_ANONYMOUSLY"/>
<anonymous enabled="false"/>
<headers>
<!--frame-options policy="SAMEORIGIN"/-->
<frame-options policy="ALLOW-FROM" strategy="static"
value="http://otherServerHost:otherServerPort" />
<!--content-security-policy policy-directives="frame-ancestors 'self'"/-->
<content-security-policy policy-directives="frame-ancestors 'self'
http://otherServerHost:otherServerPort"/>
</headers>
</http>
...
</beans:beans>
```

## 8.5 Outbound API

MashZone NextGen provides an outbound API to pass data from MashZone NextGen dashboards to an embedding system, for example, an external web application like webMethods Business Console.

See **Post data** for details.

## 8.6 Inbound API

By using iFrame MashZone NextGen can be used as a widget in external products, for example, webMethods Business Console. As embedded widget MashZone NextGen is enabled to send data via outbound API (Post data) to the embedding system and receive data via inbound API (URL selection) from the embedding system.

See Embedding MashZone NextGen in external system environments (page 52) for details.

# 9 MashZone NextGen Repositories

The MashZone NextGen Repository is the database that the MashZone NextGen Server uses to store meta-data, attributes and configuration for MashZone NextGen including:

- Artifacts (dashboards and data feeds)
- Configuration properties for the MashZone NextGen Server

If you are using the default User Repository, user and group data is also stored in the MashZone NextGen Repository.

The MashZone NextGen repository is initially installed in a Derby database suitable only for trial purposes. For proof-of-concept, development or production uses, move the repositories to a robust and compatible solution.

Configuration and administration tasks for these two repositories include:

- Move the MashZone NextGen repository to a robust database solution (page 24)
- Support International Character Sets and Locales (page 47)
- Use the Default MashZone NextGen User Repository (page 39)
- Change MashZone NextGen Repository Ports (page 50)
- Tuning the MashZone NextGen Repository Connection Pool (page 145)
- Synchronize the MashZone NextGen Repository and MashZone NextGen Server Time Zones (page 146)
- Sharing the MashZone NextGen Repository in Clustered Environments (page 110)
- Configure BigMemory Max Servers for MashZone NextGen Caching and In-Memory Stores (page 66)
- Maintenance Suggestions (page 144)

## 9.1 Maintenance Suggestions

Your existing standards for database backups, security and maintenance can be applied to the MashZone NextGen repositories. In addition, you should set up procedures to monitor or regularly manage growth for the MashZone NextGen Auditable Events table. This table tracks audit information for updates to the MashZone NextGen Repository.

You may also want to move snapshot data to a separate database to more easily manage growth and other operations for these datasets.

## 9.2 Tuning the MashZone NextGen Repository Connection Pool

In addition to basic connection configuration, you can configure the connection pools for the MashZone NextGen Repository. In many cases, you need to tune this configuration to optimize your MashZone NextGen environments.

For a complete list of connection properties, see the official Tomcat Datasource Properties.

To tune the connection pool, you update properties in the **<Resource>** element for the MashZone NextGen repository in the <MashZone NextGen installation>/apache-tomcat/conf/context.xml file and then restart MashZone NextGen to apply these changes.

### 9.2.1 Connection Pool Size Properties

| | |
|---|---|
| initialSize | The initial number of connections to create when the pool starts up. This defaults to 0. |
| maxWaitMillis | The maximum number of milliseconds that the pool will wait when no connections are available before failing. Defaults to -1 which is an indefinite wait. |

### 9.2.2 Idle Pool Connection Properties

| | |
|---|---|
| maxIdle | The maximum number of connections that can be idle without connections being released. Defaults to 20. Set this to -1 to prevent any connections being released. |
| minIdle | The minimum number of idle connections that can exist before new connections are added to the pool. This defaults to 0, indicating no new connections should be created. |
| testWhileIdle | Whether connections should be tested when idle.<br>If this is enabled, idle connections are tested using the **Validation query**. See Move the MashZone NextGen repository to a robust database solution (page 24) for more information on validation queries. |

| | |
|---|---|
| timeBetweenEvictionRunsMillis | The number of milliseconds between tests of idle connections. This defaults to -1, which prevents all idle connection testing. |
| numTestsPerEvictionRun | The number of connections to test during any idle connection test run. |
| minEvictableIdleTimeMillis | The minimum number of milliseconds that a connection can be idle before being tested for eviction. Default is 3 minutes. |

## 9.3 Synchronize the MashZone NextGen Repository and MashZone NextGen Server Time Zones

Creation and modification timestamps for artifacts and other MashZone NextGen Repository metadata can be different than times when events occurred in the MashZone NextGen Server in two cases:

- If the server hosting the MashZone NextGen Repository is located in a different time zone from the server hosting the MashZone NextGen Server
- If the time zone setting for the database hosting the MashZone NextGen Repository is set to a different time zone from the server hosting the MashZone NextGen Server

You can correct this problem by specifying a time zone in configuration for the MashZone NextGen Repository.

The instructions in this topic are specific to MySQL databases. For other types of databases, please consult documentation for that database to determine the appropriate updates.

# 10 Additional Information and Support

## 10.1 Samples, Help and Other Documentation

You can find samples, help and other documentation for MashZone NextGen in:

- MashZone NextGen User and Developer Guide: You can download MashZone NextGen documentation at **https://empower.softwareag.com/** for offline access. An Empower account is required.
- **Online Help**: is accessible in MashZone NextGen from **?** help buttons.
- **Documentation for Other Software AG Products**: is available online at **http://documentation.softwareag.com** with an Empower or from **http://techcommunity.softwareag.com/welcome-documentation** with a Software AG Tech Communities account.

## 10.2 Version and License Information

You can find version and other system information for MashZone NextGen in the Admin Console.

For system information:

1. Open the MashZone NextGen Admin Console.
2. Expand the **Server** menu section.
3. Click **System Info** and select the **Platform Information** or **System Information** tabs.
4. Click **License Info** for MashZone NextGen license information.

# 11 Legal information

## 11.1 Documentation scope

The information provided describes the settings and features as they were at the time of publishing. Since documentation and software are subject to different production cycles, the description of settings and features may differ from actual settings and features. Information about discrepancies is provided in the Release Notes that accompany the product. Please read the Release Notes and take the information into account when installing, setting up, and using the product.

If you want to install technical and/or business system functions without using the consulting services provided by Software AG, you require extensive knowledge of the system to be installed, its intended purpose, the target systems, and their various dependencies. Due to the number of platforms and interdependent hardware and software configurations, we can describe only specific installations. It is not possible to document all settings and dependencies.

When you combine various technologies, please observe the manufacturers' instructions, particularly announcements concerning releases on their Internet pages. We cannot guarantee proper functioning and installation of approved third-party systems and do not support them. Always follow the instructions provided in the installation manuals of the relevant manufacturers. If you experience difficulties, please contact the relevant manufacturer.

If you need help installing third-party systems, contact your local Software AG sales organization. Please note that this type of manufacturer-specific or customer-specific customization is not covered by the standard Software AG software maintenance agreement and can be performed only on special request and agreement.

## 11.2 Support

If you have any questions on specific installations that you cannot perform yourself, contact your local Software AG sales organization (https://www.softwareag.com/corporate/company/global/offices/default.html). To get detailed information and support, use our Web sites.

If you have a valid support contract, you can contact **Global Support ARIS** at: **+800 ARISHELP**. If this number is not supported by your telephone provider, please refer to our Global Support Contact Directory.

## ARIS COMMUNITY

Find information, expert articles, issue resolution, videos, and communication with other ARIS users. If you do not yet have an account, register at ARIS Community**.**

## PRODUCT DOCUMENTATION

You can find the product documentation on our documentation Web site.

In addition, you can also access the cloud product documentation. Navigate to the desired product and then, depending on your solution, go to **Developer Center**, **User Center** or **Documentation**.

## PRODUCT TRAINING

You can find helpful product training material on our Learning Portal.

## TECH COMMUNITY

You can collaborate with Software AG experts on our Tech Community Web site. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software AG news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories and discover additional Software AG resources.

## PRODUCT SUPPORT

Support for Software AG products is provided to licensed customers via our Empower Portal (https://empower.softwareag.com/). Many services on this portal require that you have an account. If you do not yet have one, you can request it. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Add product feature requests.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.