

# Elixir Administration Tools

Release 3.5.0



*Elixir Technology Pte Ltd*

---

# **Elixir Administration Tools: Release 3.5.0**

*Elixir* Technology Pte Ltd

Published 2014

Copyright © 2014 Elixir Technology Pte Ltd

All rights reserved.

Java is a trademark or registered trademark of Oracle Corporation in the United States and other countries. Microsoft and Windows are trademarks of Microsoft Corporation.

---

---

# Table of Contents

1. Elixir Administration Tools .....	1
Overview .....	1
System .....	1
Unused Files .....	2
Unused Sessions .....	2
Targets .....	2
Configuration .....	8
Libraries .....	9
Configuring JDBC Drivers .....	9
Universe .....	10
Universe Mapping .....	11
Universe Access .....	12
2. System Monitoring With StatsD .....	14
Overview of StatsD .....	14
Enabling StatsD .....	14
Viewing Your System's Statistics .....	15
Statistics Listing .....	17
3. Ambience - LDAP Integration .....	18
Setting up Ambience Integration With LDAP .....	18
Setting up the Apache Directory Server .....	18
Integrating Ambience With LDAP .....	18

---

## List of Figures

1.1. System .....	1
1.2. Libraries .....	9
1.3. Refresh Job Engine Jars .....	10
1.4. Universe .....	11
1.5. Add Universe .....	11
1.6. Publish Template Dashboard .....	12
1.7. Publish Template Report .....	12
1.8. Universe Access .....	12
2.1. StatsD Monitoring Protocol .....	14
2.2. Adding a Graph Panel .....	15
2.3. Adding the Metrics .....	16
2.4. Displaying the Added Statistic .....	16
2.5. Completed Dashboard .....	16

---

## List of Tables

1.1. Output Types for Report Rendering .....	3
1.2. JDBC Target Configurations .....	4
1.3. JMS Target Configurations .....	4
1.4. Mail Target Configurations .....	4
1.5. PDF Signer Target Configuration .....	5
1.6. Print Target Configurations .....	6
1.7. Repository Target Configurations .....	7
1.8. S3 Target Configurations .....	7
1.9. SFTP Target Configurations .....	8
1.10. Socket Target Configurations .....	8
1.11. Repository User Home Target Configurations .....	8
2.1. Performance Metrics .....	17

---

# Chapter 1

## Elixir Administration Tools

---

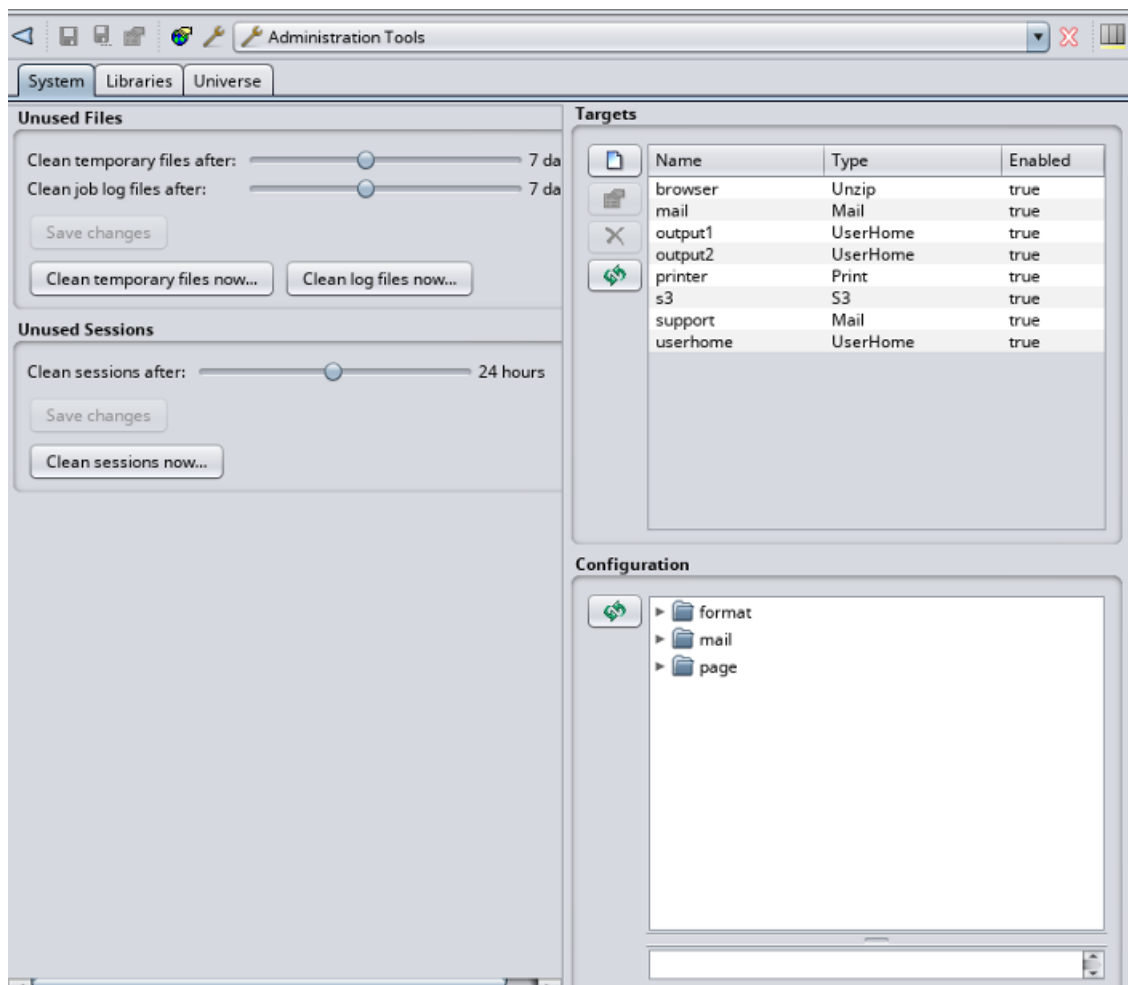
### Overview

On clicking the Admin Tools... button on the Action Bar, you will see the Administration Tools panel. These tools include System, Libraries, and Universe.

### System

The System page allows you to clean up unwanted log and temporary files, set targets and configure formats such as date and time.

**Figure 1.1. System**



## Unused Files

Allows you to set the duration after which unused temporary and log files are automatically deleted.

The default duration is set to 7 days.

You can also elect to clean up the temporary and log files immediately, by clicking **Clean temporary files now** and **Clean log files now**.

## Unused Sessions

Allows you to set the duration after which unused sessions are automatically deleted.

The default duration is set to 24 hours.

You can also elect to clean up the unused sessions immediately, by clicking **Clean sessions now**.

## Targets

After a job finishes, a target may pick up the result. Different targets may be chained for further processing. Target processing is also a task run by the Job Engine.

## Target Constants

If you find yourself typing a string repetitively when configuring targets, you can define that string as a constant. Then you can refer to that constant in target property values like `${constant-name}`. Target constants can be enabled/disabled. Only enabled constants can be used in target configurations. You can define multiple constants with the same name, but only one of them can be enabled at any time.

## Target List

- **Target Creation/Update/Deletion**  
All manipulations can be done in the Target Wizard. For most of the targets, configuration is simple. You need to provide a name, enable the target and provide values for required target properties.
- **Status**  
Targets can also be enabled/disabled. Only enabled targets can be used in RenderReport task. You can define multiple targets with the same name, but only one of them can be enabled. If you make one target enabled, the rest of the targets with the same name will be disabled automatically.  
  
Targets can be available/unavailable to anonymous users. When the **Available to anonymous users** check box is selected, the target is accessible in anonymous mode. By default, only the "browser" target is available.
- **Target Properties and Parameters**  
A target requires certain properties such as file name, folder name, port number and so on.  
  
When editing a target, the Target Wizard lists all required properties for the target. An administrator has several options when configuring those property values:
  - Provision of an exact string value for some properties.
  - Reference to target constants.
  - Definition of parameters in some properties if they should be provided by end users. For example, the parameter in property "filename" can be defined as `"${file#report}_${date}"`. End users should provide values for those parameters when invoking targets.

- **Output Types for Report Rendering**  
A target provides various output types such as CSV, HTML, PDF, Microsoft formats and so on.

After the properties are set, the Target Wizard enables you to choose among different output types for report rendering purpose. You can choose multiple types at a time for convenience. You can also print the report directly.

**Table 1.1. Output Types for Report Rendering**

CSV	Glnt
HTML	Line Printer
Logical RML	Logical RML Tree
Microsoft DOCX	Microsoft Excel
Microsoft PPTX	ODF Spreadsheet
PCL	PDF
Postscript	Print
Rich Text Format	Simple HTML
XML	Zipped Bitmap
Zipped Jpeg	Zipped Png
Zipped Svg	Zipped Tiff
Zipped Wbmp	

## Create New Target

When you create a new target using the Target Wizard, you should fill in values, for example parameters, into the property fields. Nested parameters are not supported. For example, you cannot specify a parameter like `${file##${reportname}}`.

## JDBC Target

A JDBC Target allows reports to be written directly into a database. This is useful if you have some subsequent program to pick them up or otherwise act on them - for example a document management system. Each report is written as a record into a specific table in the database. The report data itself is stored as a BLOB. Before you can use the JDBC target, you need to set up a database with a table that has the correct schema to accept a report file. An example as shown:

```
CREATE TABLE JOBOUTPUT (
  id INTEGER NOT NULL GENERATED ALWAYS AS
  IDENTITY (START WITH 1, INCREMENT BY 1),
  name VARCHAR(256) NOT NULL,
  lastModified BIGINT NOT NULL,
  content BLOB NOT NULL,
  CONSTRAINT JOBOUTPUT_PK PRIMARY KEY(id) )
```

Once the database is setup, configuration will write into a table called JobOutput in the Derby database that is built into the Elixir Repertoire Server:



**Table 1.2. JDBC Target Configurations**

Name	Value
Driver	org.apache.derby.jdbc.EmbeddedDriver
URL	jdbc:derby:/home/.../jdbctarget
Table	\${update table##JobOutput}
User	\${userid##Enter your userid}
Password	\${password##Enter your password}
Filename	\${filename##Job_Output_Report}
Overwrite	true

## JMS Target

A JMS Target can be used for asynchronous messaging. JMS applications can use job messages as a form of managed request/response processing, to give remote feedback to the users on the outcome of their send operations and the fate of their messages. Examples of job messages are Exception, Expiration, Confirm on arrival (COA), Confirm on delivery (COD), etc.

**Table 1.3. JMS Target Configurations**

Name	Value
Destination	RQueue
User	\${userid##Enter your userid}
Password	\${password##Enter your password}
Filename	\${filename##Pet_Store_User_Accounts_Report}
Reply required	true
Timeout	\${reply timeout in secs##30}
Reply success keyword	\${reply success keyword##OK}
Reply success pattern	\${reply success pattern##^.*OK.*\$}

## Mail Target

A Mail Target allows the output to be sent by email. There are a number of parameters to specify, but remember that you can use substitutions to avoid hard-coding those that you decide need to be flexible. The report will be sent as an attachment by email, so you can choose the render format you prefer.

**Table 1.4. Mail Target Configurations**

Name	Value
To	\${to##sam@elixirtech.com}
SMTP host	elixir.aspirin
From	\${from##susan@elixirtech.com}
Cc	\${cc##bob@elixirtech.com}
Subject	\${subject##Pet Store User Accounts Report From Elixir Server}
Message	\${message##Your report is attached.}
Filename	\${filename##Pet_Store_User_Accounts_Report}

## PDF Signer Target

PDF Signer Target is used when a PDF output needs to be signed digitally. It prints a "signature" in a PDF file when the file is rendered.

**Table 1.5. PDF Signer Target Configuration**

Name	Value
Keystore	<code>\${key.url##C:/Java/jre7/bin}</code>
Keystore type	<code>\${keystore-type}</code>
Store encrypted password	false
Store password	<code>\${key-password##keyStorePassword}</code>
Key alias	<code>\${key-alias}</code>
Signer appearance	<code>\${signer-appearance##self-signed}</code>
Info reason	<code>\${info-reason##Reason for signature}</code>
Info location	<code>\${info-location##bottom-left}</code>
Sign width	<code>\${sign-width##100}</code>
Sign height	<code>\${sign-height##60}</code>
Sign page	sign-last-page
Sign image	<code>\${signature-image-url##D:/image/logo.png}</code>
Certification level	<code>\${certificate-level##certificate-no-changes}</code>
Filename	Specify file name here

## PDF Signer Target Properties

### Signer appearance

There are two types of signing mode. They are *self-sign* and *wincer-sign*.

*self-sign* signer can be generated using Java key generator (keytool.exe). An example of a signature command is as follows :

```
keytool -genkey -keyalg RSA -alias QAkey -keypass
mypassword
    -keystore keystore.ks -dname "cn=My Key Name, c=SG"
```

*wincer-sign* is recommended if higher security is required. Certificate obtained is installed to the web browser and needs to be exported to be used in PDF signing.

### Keystore type

For *self-sign*, no keystore type is required. For *wincer-sign*, the keystore type will depend on the respective vendor. For example, a VeriSign keystore type will be pks12.

### Keystore

The directory of the keystore is entered here. *repository* is prohibited for this parameter.

### Key alias

This parameter is optional. It is to define the alias used by the keystore file.

### Store password

*key-password* is the password entered when creating the keystore file. The password can either be in encrypted or unencrypted form.

### Store encrypted password

This parameter holds a boolean value, `true` if the key password is encrypted and `false` if otherwise.

### Sign page

This parameter is to specify the page that the signature is placed. *sign-no-page* will mean that the signature will not be visible. *sign-first-page* and *sign-last-page* implies that the signature will be placed on the first and last page of the PDF file respectively. If you wish to place the signature on a specific page, enter the page number as the value for the parameter. For example, `sign-page##5`, for placing the signature on the 5th page of the PDF document.

### Sign width and Sign height

These two parameters are for user to specify the height and width of the signature rectangle size.

### Sign image

The directory of the image that is to be used with the signature. When no value is entered for this parameter, there will still be a PDF Signature (self-signed) automatically generated.

### Info reason and Info location

The reason for placing a signature in the PDF document and the position of the reason.

### Certification level

<i>certificate-not</i>	has no visible signature in the PDF document. The document is digitally counter signed.
<i>certificate-no-changes</i>	displays the signature in the PDF document and no changes can be made to the document.
<i>certificate-form-filling</i>	also displays the signature in the PDF document, but only the filling in of forms, signing and page adding is allowed for the document.
<i>certificate-form-filling-annotation</i>	displays the signature in the PDF document, but only commenting, form fill-in, signing and page adding is allowed.

## Print Target

A Print Target allows you to send a report to a named printer. The only option is the name of the printer. If you have multiple alternate printers, you could use a separate target for each printer so that you could control access by different groups. In most cases, the printer names will be fixed and not include substitutions. You can also leave the printer name blank as it will route automatically to the default printer defined on the server.

**Table 1.6. Print Target Configurations**

Name	Value
Printer name	Canon iR C3220 PCL5c

## Repository Target

A Repository Target writes the report to the filesystems. You can identify a target folder in the repository and provided it is writable, files will be written there. This works regardless of whether the target

filesystem is of type local, secure or dbfs. You should use Repository Targets when you want to allow users to view the reports through their browser as the repository will automatically update to show the latest files.

During the configuration, the "folder" property must refer the "dir##" parameter to an existing target filesystem in the repository. The "folder##" parameter need not as it will create a new folder after its name if it is not found in the repository.

**Table 1.7. Repository Target Configurations**

Name	Value
Folder	\${dir##ElixirSamples}/\${folder##repository}
Filename	\${filename##Pet_Store_User_Accounts_Report}
Overwrite	true

### S3 Target

A S3 Target allows the report to be transferred to a user's Amazon S3 bucket. The available parameters are bucket name, access key, secret key, the folder name in which to transfer, the expiry time of the file (if any), the filename to name the transferred file and the presigned URL protocol.

**Table 1.8. S3 Target Configurations**

Name	Value
Available to Anonymous Users	If enabled, anonymous users can access the file. Else, only those users who have logged into S3 can access the file.
Bucket	The name of the S3 bucket in which the file is to be stored.
Access Key	Enter your Amazon access key.
Secret Key	Enter your Amazon secret key.
Folder	The folder in which to store the file.
Expiry	The number of days after which the file cannot be accessed anymore.
Filename	\${filename##Pet_Store_User_Accounts_Report}
Presigned URL Protocol	A signed URL is a URL which is valid for a specific time period. Its purpose is to share the pages that have time sensitive information, or when you want to share a file with specific people alone. Such information becomes invalid once the specified time period has passed.

### SFTP Target

A SFTP Target allows the report to be transferred to a user's secured FTP Server. The available parameters are user, password, host, port, dir and filename. The port is optional and will default to 22, which is the default SFTP port, if not specified.

The parameter in "dir" property must be an existing directory found in the target ftp server.

**Table 1.9. SFTP Target Configurations**

Name	Value
Host	\${sftp host##domain_name.com}
Port	\${port##22}
Directory	\${dir##dept1}/\${folder##Pet_Store}
Filename	\${filename##Pet_Store_User_Accounts_Report}
User	\${userid##Enter userid to access sftp client}
Password	\${password##Enter password to access sftp client}

## Socket Target

A Socket Target sends the report to a program which is listening, typically on another machine. For example, a program can be written that listens on a company.com port 6000 and writes any data it receives to a database, or to a fax etc. It is up to the receiving program what it does with the data. The server opens a connection to the listening program, using the host and port information required by the socket target and streams the data across the network to the listening socket.

**Table 1.10. Socket Target Configurations**

Name	Value
Host	company.com
Port	6000

## Unzip Target

This target enables you to specify a file name for unzipping purpose. You can enable/disable this target, and set this target available/unavailable to anonymous users.

## User Home Target

A Repository User Home Target writes the report to the invoking user's folder in the repository. Users can share jobs and they can keep separate output without overwriting one another. In the example, once the user "Jon" signs in to the repository and runs the job, the report will be written to /User/Jon/Pet\_Store.

**Table 1.11. Repository User Home Target Configurations**

Name	Value
Folder	\${folder##Pet_Store}
Filename	\${filename##Pet_Store_User_Accounts_Report}
Overwrite	true

# Configuration

This allows you to perform common configurations that will apply across all Ambience modules. For example, you can set the uniform formats for date, datepicker, time, timepicker and timestamp. When a date is displayed by Java, it will use the /Configuration/format/date/ settings. When JavaScript is used, it will use the /Configuration/format/datepicker settings by default, unless there is a local override. Web-based modules will reflect the changes quickly, while other modules may show the changes after they are restarted.

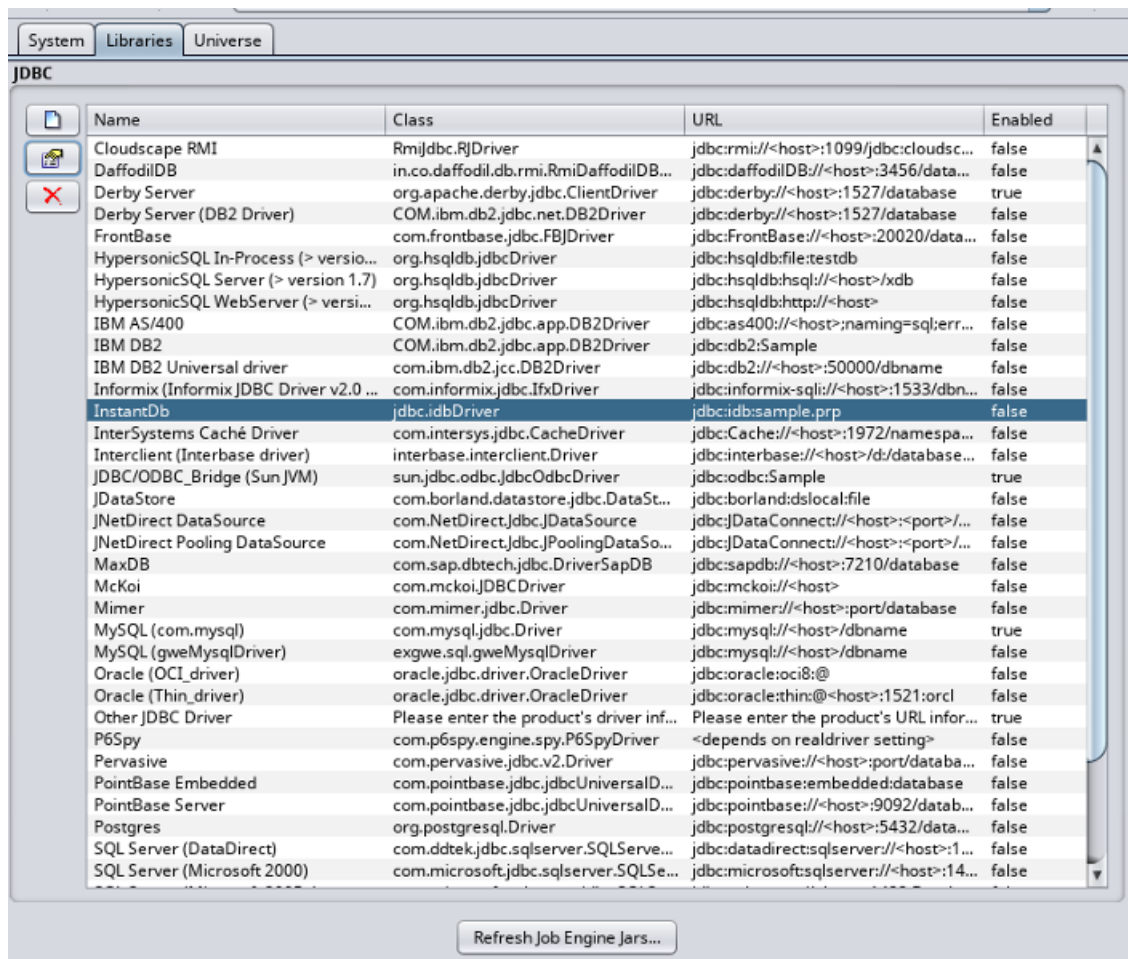
Right-click **Configuration > format > date, datepicker, time, timepicker** or **timestamp**, the following menu option will display:

- **Edit contents...:** This enables you to edit the contents of the current node. You can also validate the XML.

## Libraries

Libraries include configuration information for a full range of JDBC drivers. Repertoire 9 enables you to configure JDBC drivers more easily. Any JDBC drivers that require native components will usually require the Ambience system to be stopped and restarted. Currently, most drivers are JDBC-Net pure Java drivers or native protocol pure Java drivers. If there is no native code, then you can transfer the JAR files to the job engine machines by putting them in /Public/lib in the Repository Tree and clicking the Refresh Job Engine Jars... button. This will cause the job engines to stop at the end of their current task. The Process Managers will then start them up again, with the updated classpath (including any new JDBC drivers).

**Figure 1.2. Libraries**



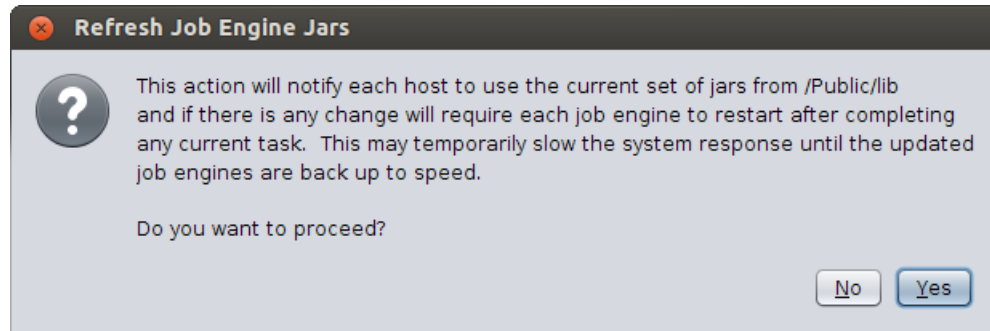
## Configuring JDBC Drivers

To illustrate the JDBC driver configuration procedure, here are the steps:

1. Click the Administration Tools... button on the workspace toolbar. Click the Libraries tab. The JDBC drivers display.

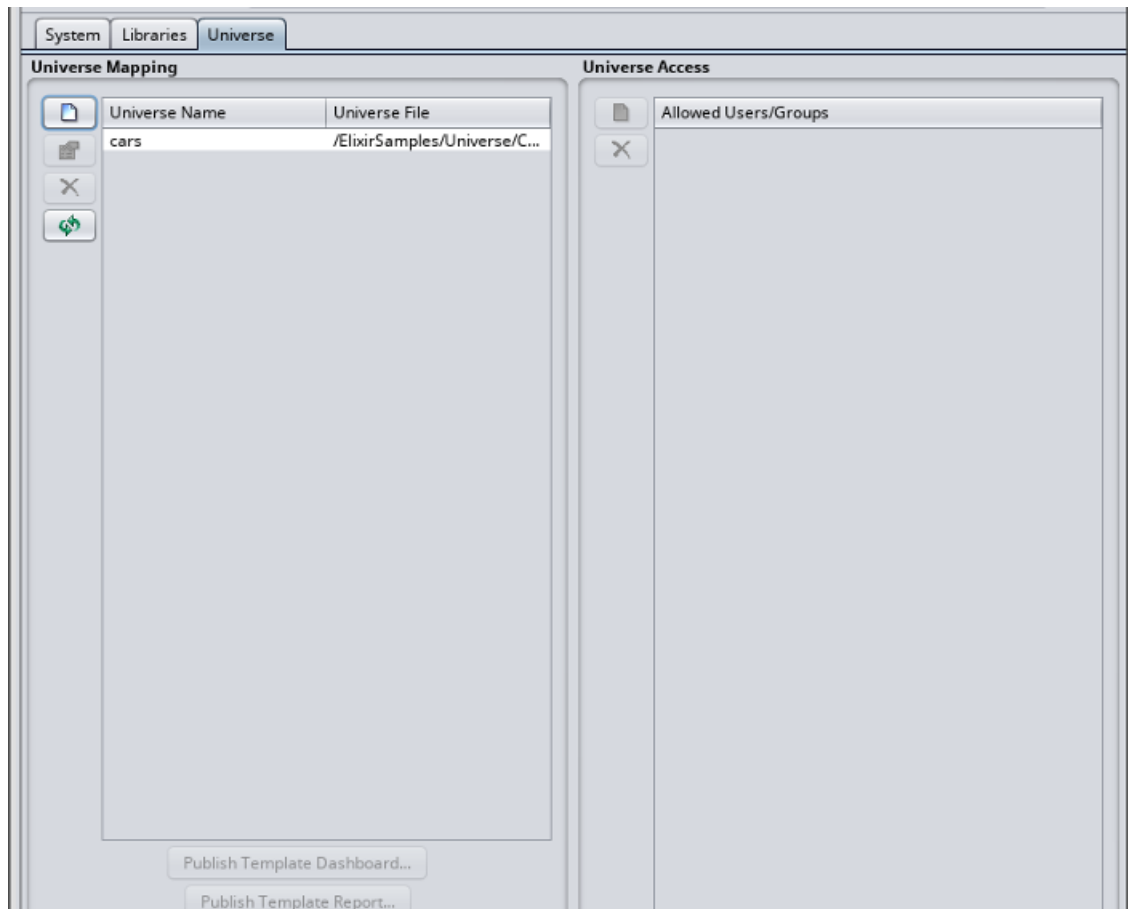
2. Find and select the desired driver from the list. Click the Edit button. The JDBC Driver dialog displays. Fill in the host machine name to replace <host> in the URL field. Leave the class information as it is.
3. Select the Enabled checkbox and click OK.
4. If there is no native code for the driver, download the driver's JAR file from the vendor's Web site. Copy the driver's JAR file into the /Public/lib folder in the Repository Tree. Click the Refresh Job Engine Jars... button. A message may pop up, as shown in [Figure 1.3, "Refresh Job Engine Jars"](#):

**Figure 1.3. Refresh Job Engine Jars**



## Universe

Through the Universe interface, you can map a Universe name with its path, publish template dashboard or report, and define which users and groups will be able to access the Universe.

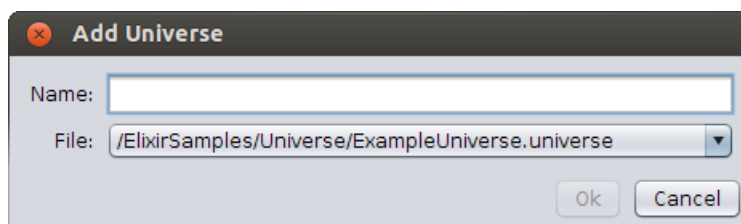
**Figure 1.4. Universe**

## Universe Mapping

A Universe can exist anywhere in the Repository, but are only accessible if its path is mapped to a Universe name. Only administrators can set or alter the Universe mapping.

Complete the following steps to map a Universe:

### Figure 1.5. Add Universe



1. In the Administration Tools, click the **Universe** tab.
2. In the Universe Mapping pane, click the **Add Universe** button.
3. In the Add Universe window, enter a name and select the desired universe from the File list.
4. Click OK. The Universe name and path are successfully mapped.



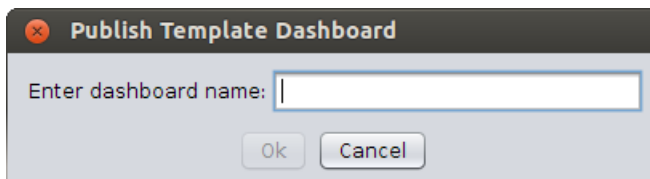
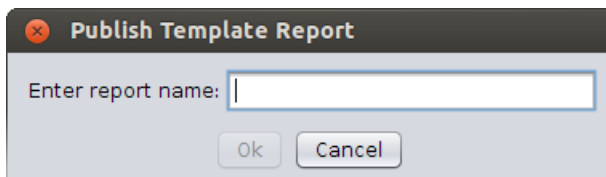
**Note**

Unwanted Universe(s) can be easily removed from the Universe Mapping pane by a mouse click.

For a Repository Universe, once its name and path have been mapped, the name will display as an option within Ad-hoc Dashboard. After setting the Primary Key, you can view the Ad-hoc Dashboard via the following URL. Fill in your domain, host machine and dashboard names to proceed:

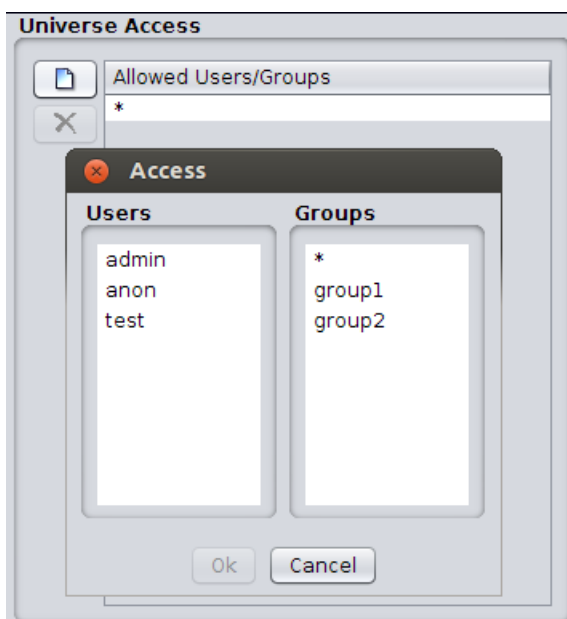
```
http://<host>:8080/elx/do/<domain>/ei/accolades/  
dashboard/Sample/  
Demo/Ad%20Hoc/Dashboard/<dashboard>.dashboard?mode=file
```

You also have the option to publish template dashboard or report, as shown in the figures below:

**Figure 1.6. Publish Template Dashboard****Figure 1.7. Publish Template Report**

## Universe Access

Complete the following steps to control the access to the Universe:

**Figure 1.8. Universe Access**

1. In the Administration Tools, click the **Universe** tab.
2. In the Universe Access pane, click the **Add Access** button.
3. In the Access window, select the desired users/groups from the list.
4. Click OK. The access rights are successfully assigned.

Only users who can read or write the Universe file can open it to view and edit the contents. However, those users in the Access list can only use the Universe, without being able to open or edit the Universe file. The Universe service will access the file for those users.

---

# Chapter 2

## System Monitoring With StatsD

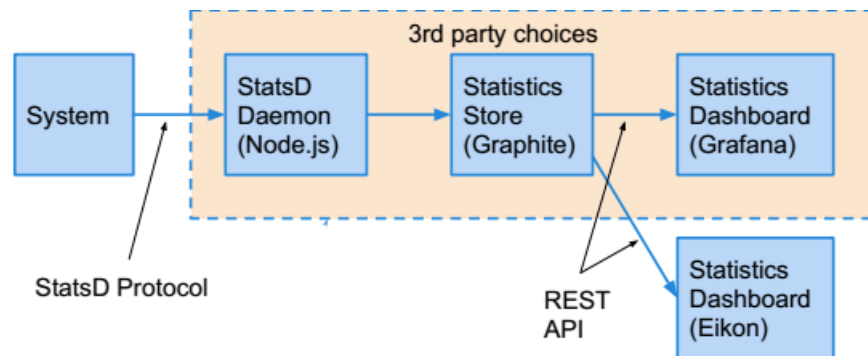
---

### Overview of StatsD

You can install StatsD to monitor the performance of your system in real time.

Figure 2.1, “StatsD Monitoring Protocol” shows a simple flow chart that depicts the StatsD monitoring protocol.

**Figure 2.1. StatsD Monitoring Protocol**



StatsD is a fast, lightweight system monitor and can use many third party tools for storing the statistics it collects (e.g. Graphite) and for displaying the statistics as graphs (for e.g. using Grafana).

Examples of metrics that StatsD can monitor include:

- Number of report pages rendered
- Number of JDBC rows accessed
- Servlet response time
- Number of Job Engines
- System RAM and CPU loads

For more information on StatsD, see <https://github.com/etsy/statsd> [<https://github.com/etsy/statsd/>].

### Enabling StatsD

To enable StatsD monitoring, add the following line to `etc/application.conf`:

```
elixir.statsd.enabled = true
```

## Viewing Your System's Statistics

If you do not already have a StatsD monitoring solution, you can install a pre-packaged, easy to use, uncomplicated docker image that contains StatsD, Graphite, Grafana and a Kamon Dashboard, all set up, from <https://github.com/kamon-io/docker-grafana-graphite>.

If you want to run this image, ensure that your system is 64-bit.

To add the statistical parameters to the dashboard, and to view the graphs:

1. Start the docker image, open your browser and navigate to `localhost` to load the Grafana home page.
2. Create a new dashboard.
3. Click the `Configure Row` icon and then add a new graph panel.
4. Add a title for the panel as shown in [Figure 2.2, “Adding a Graph Panel”](#).

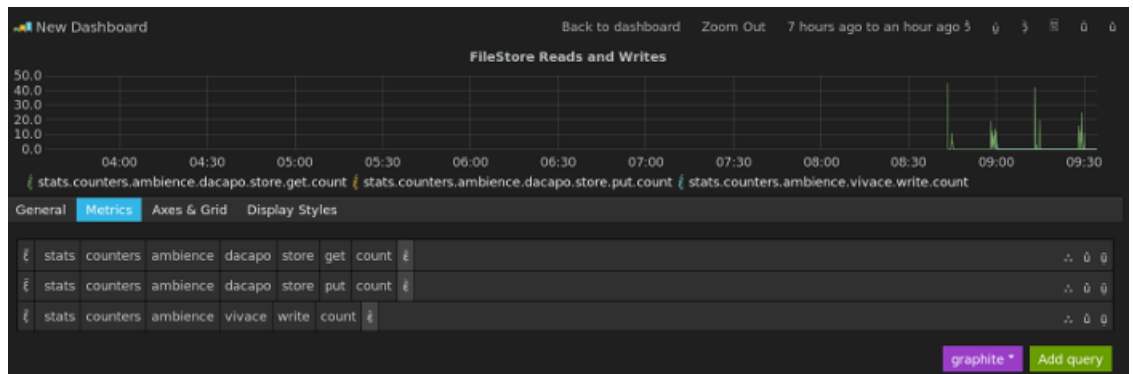
**Figure 2.2. Adding a Graph Panel**



5. Click **Add Panel** to add the panel to the dashboard.
6. Click **Close** to return to the dashboard.
7. Click the `Expand Row` icon to expand your newly added row.
8. Click the row title and select `Edit`.
9. From the `Metrics` tab, add the statistic to be retrieved and displayed, as shown by the example in [Figure 2.3, “Adding the Metrics”](#).

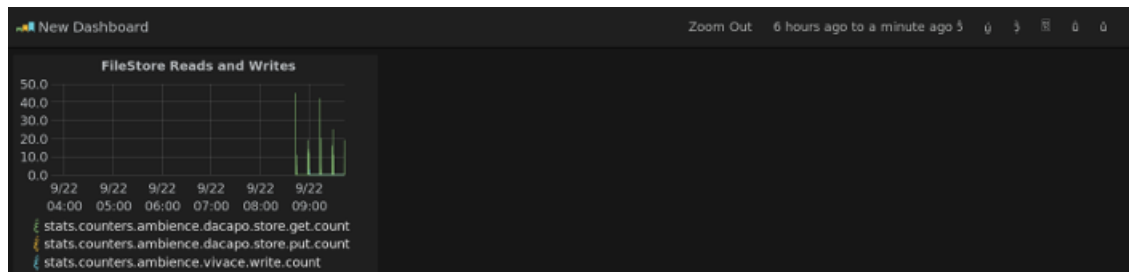
Note that as you select each component of the statistic, the system filters and displays only the appropriate choices for the next component.

**Figure 2.3. Adding the Metrics**



10. Click the **Back to Dashboard** link on the top, to return to your dashboard. The added statistic is displayed, as shown by the example in Figure 2.4, “Displaying the Added Statistic”.

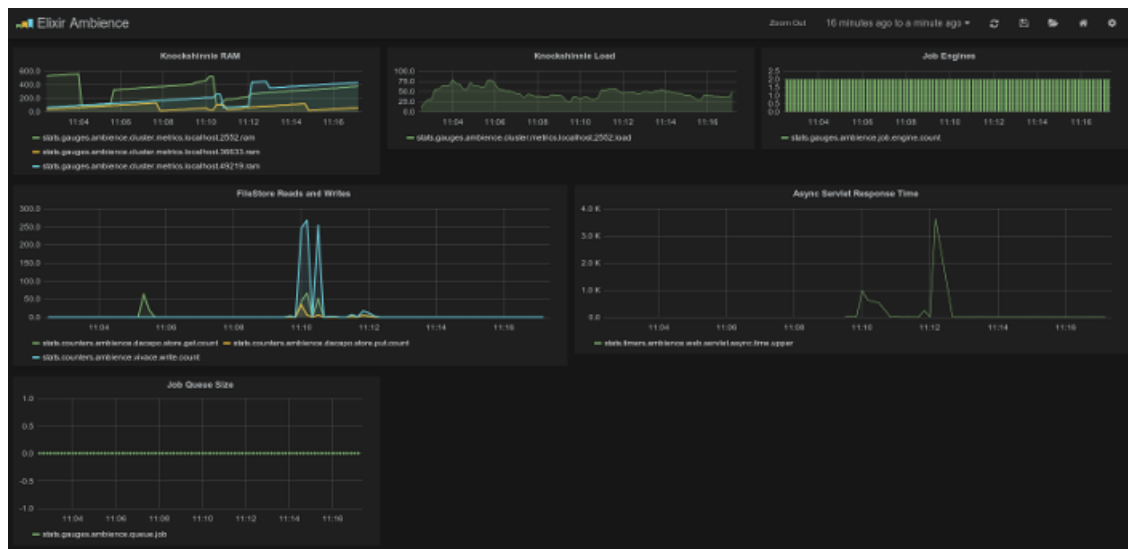
**Figure 2.4. Displaying the Added Statistic**



11. Click **Add a Row** and repeat the process to add a new statistic to the dashboard.

An example of a completed dashboard is shown in Figure 2.5, “Completed Dashboard”.

**Figure 2.5. Completed Dashboard**



## Statistics Listing

The following table lists the performance metrics that can be monitored.

**Table 2.1. Performance Metrics**

Parameter	Description
stats.counters.ambience.dacapo.indexer.file	Counts the number of files that have been indexed.
stats.counters.ambience.dacapo.store.get	Counts the number of files that <code>dacapo</code> has read from the underlying file store.
stats.counters.ambience.dacapo.store.put	Counts the number of files that <code>dacapo</code> has written to the underlying file store.
stats.counters.ambience.vivace.write.count	Counts the number of files that <code>vivace</code> has written to the disk.
stats.counters.ambience.vivace.write.rate	Denotes the rate of change. For example, 10 nodes per second.
stats.gauges.ambience.cluster.metrics.localhost.[numbers].load	Measures the CPU load of the machine running JVM on port [numbers].
stats.gauges.ambience.cluster.metrics.localhost.[numbers].ram	Measures the amount of RAM used by the machine running JVM on port [numbers].
stats.timers.ambience.job.infowatcher.time.[metric]	Measures the time taken for the job engine to process each request.

Counts are not cumulative - for example if six nodes are written in 10 seconds and then eight in the next ten seconds, you will get a chart showing 6, 8. Ten seconds is the default StatsD count interval.

---

# Chapter 3

## Ambience - LDAP Integration

---

---

### Setting up Ambience Integration With LDAP

The following **example** procedure demonstrates how to integrate Ambience with LDAP. Once integrated, you can authenticate LDAP users to Ambience.

The following procedure has been tested using Apache Directory Server. **Adapt the steps to your infrastructure**.

### Setting up the Apache Directory Server

1. Set up Apache Directory Server and start it.
2. From Apache Directory Studio, set the following properties:
  - New Connection Port: 10389
  - Bind DN or User: uid=admin,ou=system
  - Bind Password: secret
3. To add a new partition, click the connection and select **Open Connection**.
4. Click **Advanced partitions configuration**.
5. Add partition: ID=elixirbase Suffix=dc=elixirbase,dc=com.
6. Save the changes.
7. Restart Apache Directory Server.
8. Refresh the LDAP tree. Delete the node: dc=elixirbase,dc=com as the LDIF file includes it.
9. Click **File > Import > LDIF** and import elixirbase.ldif which is in the misc directory of the Elixir Ambience build.

elixirbase.ldif is now loaded into Apache Directory Server for LDAP queries.

### Integrating Ambience With LDAP

1. You need to change the Ambience run script. Copy run-server.sh to run-ldap.sh.
2. Add the following to the Java startup:  
-Delixir.usergroup.factory=com.elixirtech.arch.usergroup.ldap.LDAPUser-GroupDomainFactory

Alternatively, you can add the following line to /etc/application.conf, if you prefer:

```
elixir.usergroup.factory=com.elixirtech.arch.usergroup.ldap.LDAPUser-GroupDomainFactory
```

3. Make sure your Apache Directory Server is running and launch Ambience using `run-ldap.sh`.
4. Login to the Ambience Domain Manager.



If your LDAP server is running on a different host or port, there will be an error in the log. Navigate to **Configuration > module > usergroupdb > ldap** and alter the host and port values accordingly.

5. Navigate to **Configuration > module > usergroupdb > function > SignIn**. Edit the users array at the end of the configuration to include `aimi` (an administrator): `"users": [ "aimi" ]`.
6. Login to Ambience as user: `aimi` with password: `aimi`.
7. Navigate to **Admin > Access Matrix**. Here you can turn on `SignIn` permissions for other users (subject to licence limits). Turn on access for user: `david`, then log off and verify you can login as user: `david` (with password: `david`). `David` is not an admin user (those in group `Atlas` are admin users), so you will need to login as user: `aimi` to make further admin changes.
8. Finally as user: `aimi`, move the contents of `/User/admin/cron` into `/User/aimi/cron`, as otherwise the scheduler will attempt to run the jobs as `admin` when that user exists in LDAP. You can then delete the default `admin` and `test` user folders.
9. Once you have gone through these steps to verify that your LDAP system is working properly, use the Domain Manager to look into the LDAP configuration. Navigate to **Configuration > module > usergroupdb > ldap**. The configuration reads as follows:

```
{ "version": 1,
  "connectionFactory": "com.elixirtech. \
arch.usergroup.ldap.DefaultLDAPConnectionFactory",
  "cxtFactory": "com.sun.jndi.ldap.LdapCtxFactory",
  "host": "localhost",
  "port": 10389,
  "protocol": "default",
  "method": "simple",
  "user": "uid=admin,ou=system",
  "passwordEncrypted": "8Oz2e9XJ+Grd396E4QK91Q==" }
```

Edit the host and the port as needed to connect to a different server.

The user and password here represent LDAP credentials to read all users and groups.

10. There are several sub-configurations for `DefaultLDAPConnectorFactory` (other custom connector factories may vary). Look at the contents of the sample LDIF to help understand these.

`users` contains:

```
{ "name": "ou=users,dc=elixirbase,dc=com",
  "mode": "one-level", "returnAttr": "uid",
  "focus": { "left": "uid" } }
```



This indicates the location where the user information is stored in LDAP - one level below `ou=users,dc=elixirbase,dc=com`. The user ID is stored as `uid` and you must focus on this ID to get your set of user names.

groups contains:

```
{"name": "ou=groups,dc=elixirbase,dc=com",  
"mode": "one-level", "returnAttr": "roleOccupant",  
"focus": {"left": "cn", "right": "uid"}}
```

This is similar to `users`, but now we want to not only get the groups, but the users assigned to those groups. In a Map structure this is `[Group-> List[User]]`.

The groups are found one-level below `ou=groups,dc=elixirbase,dc=com` and the users are found in the `roleOccupant` attributes. Focus your attention on `cn` (which is the group name) and the `uid` (which is the user name). This gives us `[cn-> List[uid]]` (note the left and right positions).

Finally `adminGroups` contains:

```
{"name": "ou=groups,dc=elixirbase,dc=com", "mode": "subtree",  
"filter": "cn=Atlas", "returnAttr": "cn", "focus": {"left": "cn"}}
```

This filters all the groups, selecting the one whose name is `Atlas` and returning the `cn` of that group. This makes `Atlas` the admin group and anyone who is a member of `Atlas` is considered an administrator.

This sample represents a typical layout of users and groups in LDAP. However, some systems follow different structures.

If the particular LDAP cannot be modelled using these three query structures, then a custom LDAP mechanism can be included by altering:

```
"connectionFactory": "com.elixirtech.arch.usergroup.ldap.DefaultLDAP-  
ConnectionFactory"
```

to point to a custom implementation.

Contact Elixir Professional Services to get your custom implementation developed.